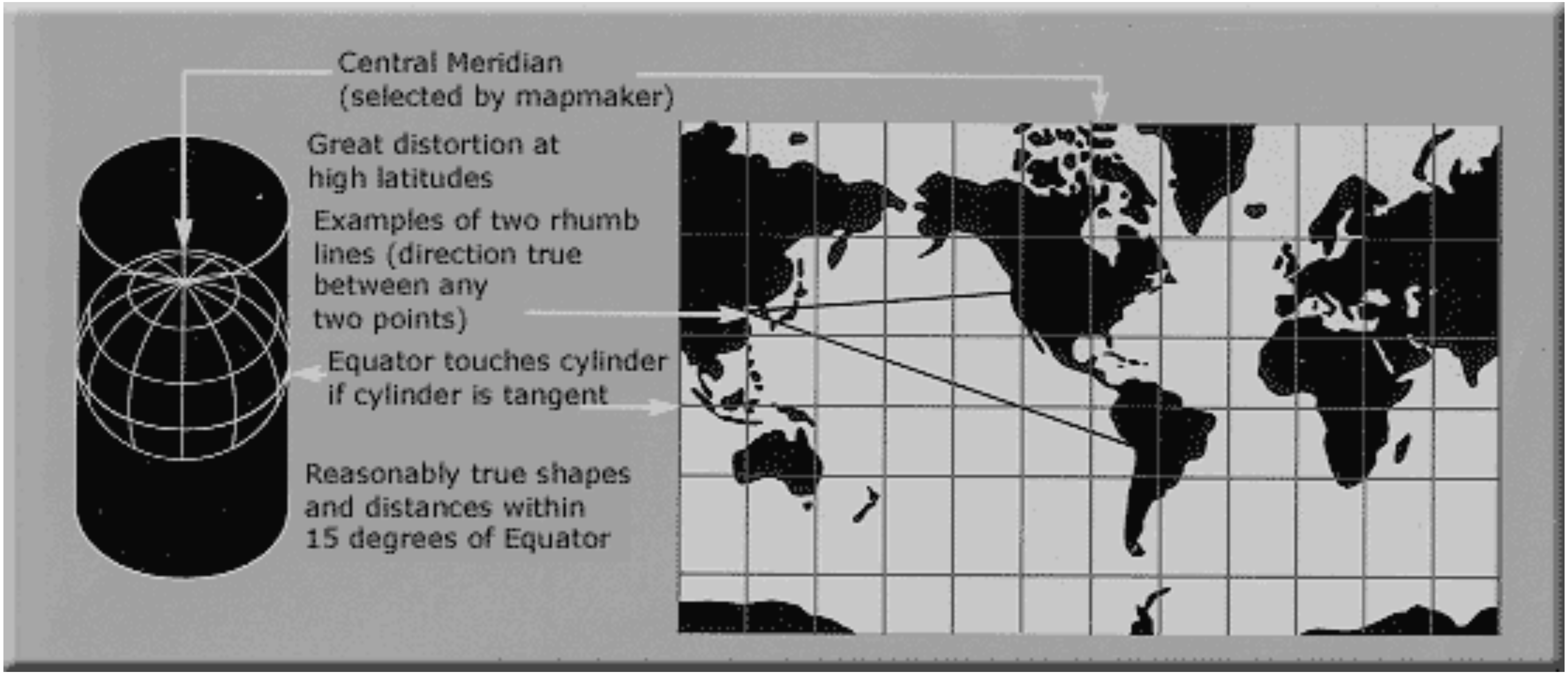
Data Visualization (CIS 490/680)

Trees & Design

Dr. David Koop

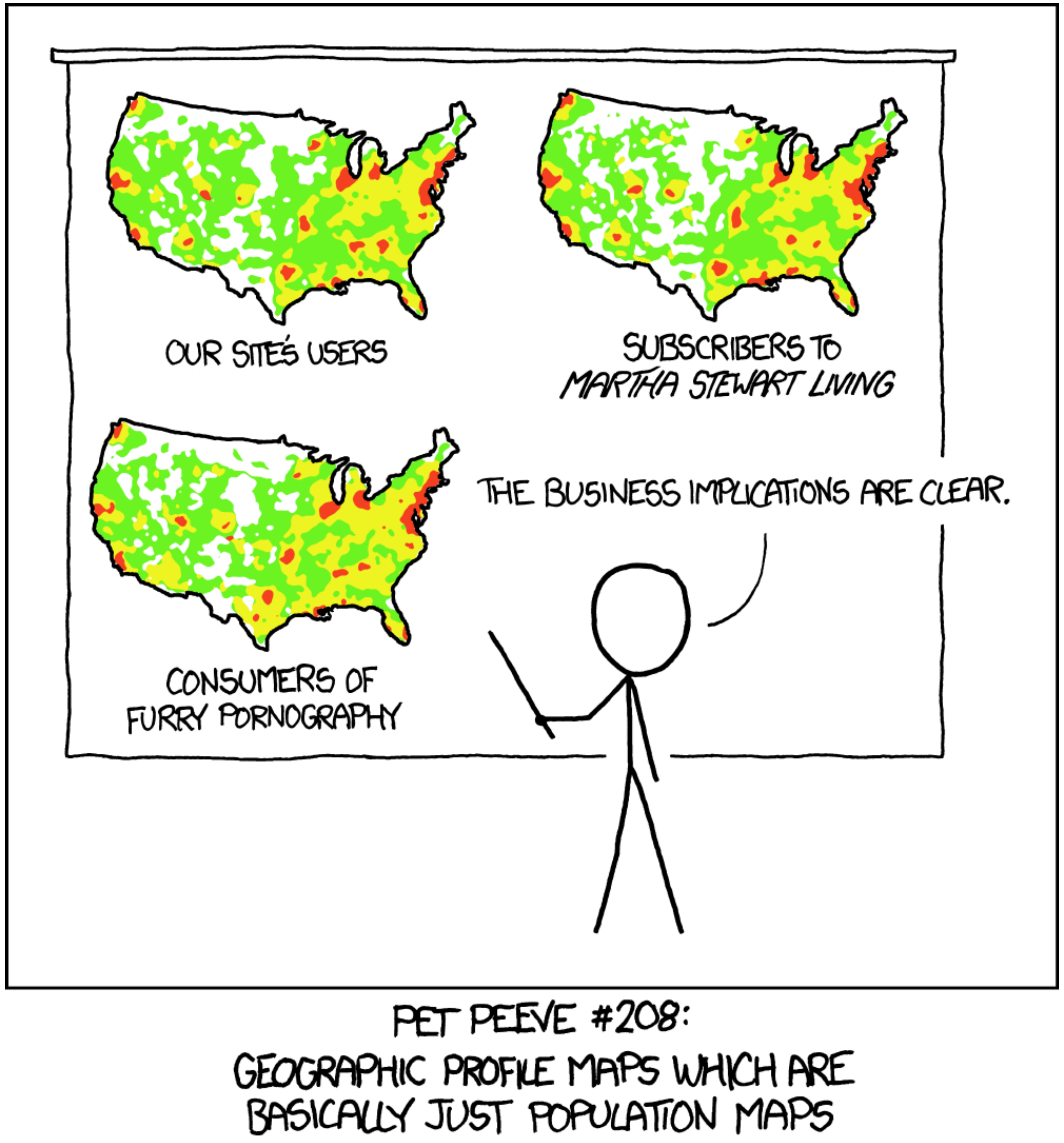
What are the ingredients for a geospatial visualization?

Map Projection + Position Data



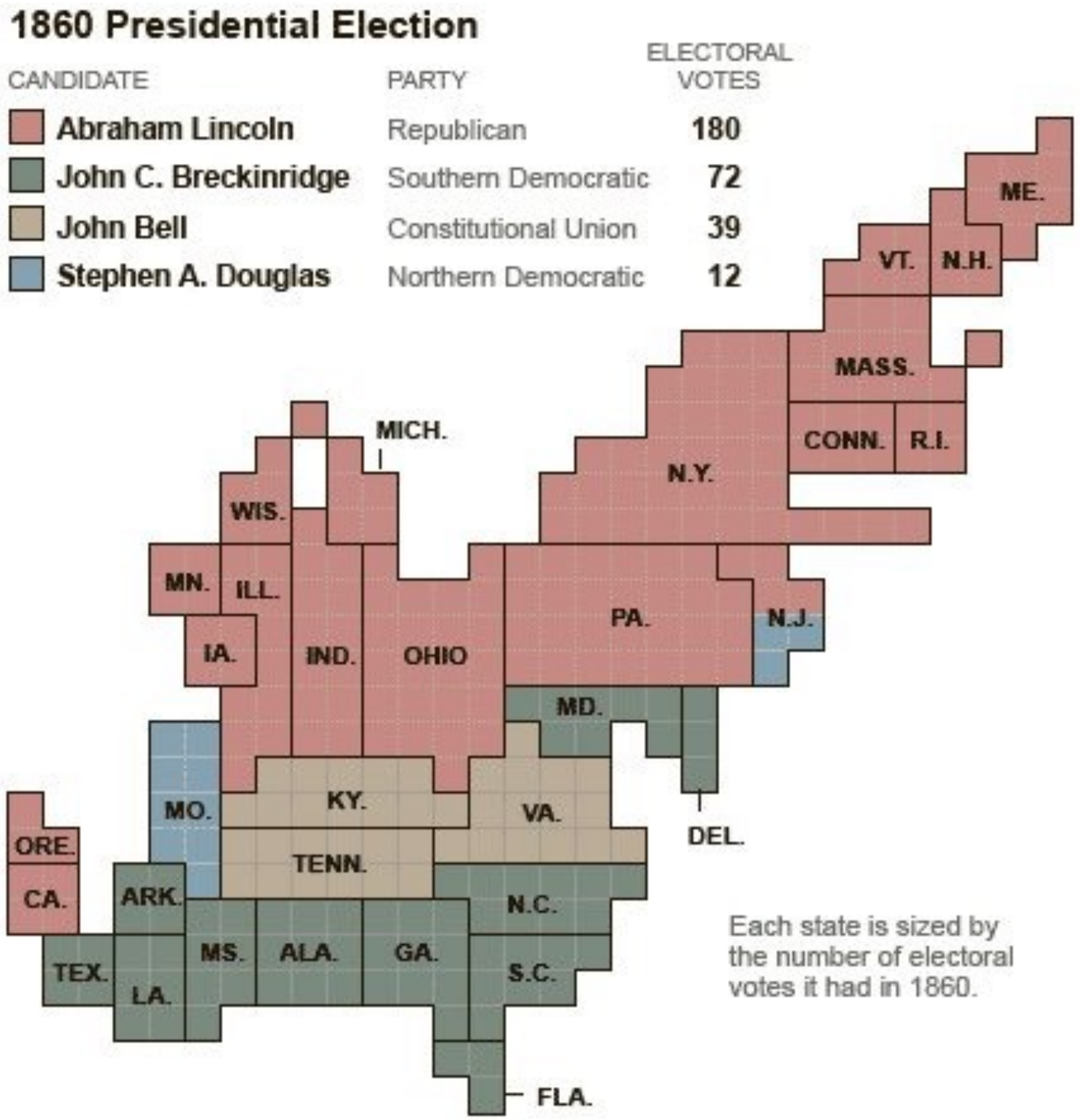
[USGS Map Projections]

Don't Just Create Population Maps!



[xkcd]

Cartograms



- Data: geographic geometry data & **two** quantitative attributes (one part-of-whole)
- Derived data: new geometry derived from the part-of-whole attribute
- Tasks: trends, comparisons, part-of-whole
- How: area marks from derived geometry, color hue/saturation/luminance
- Scalability: thousands of regions
- Design choices:
  - Colormap
  - Geometric deformation

[New York Times]

Networks

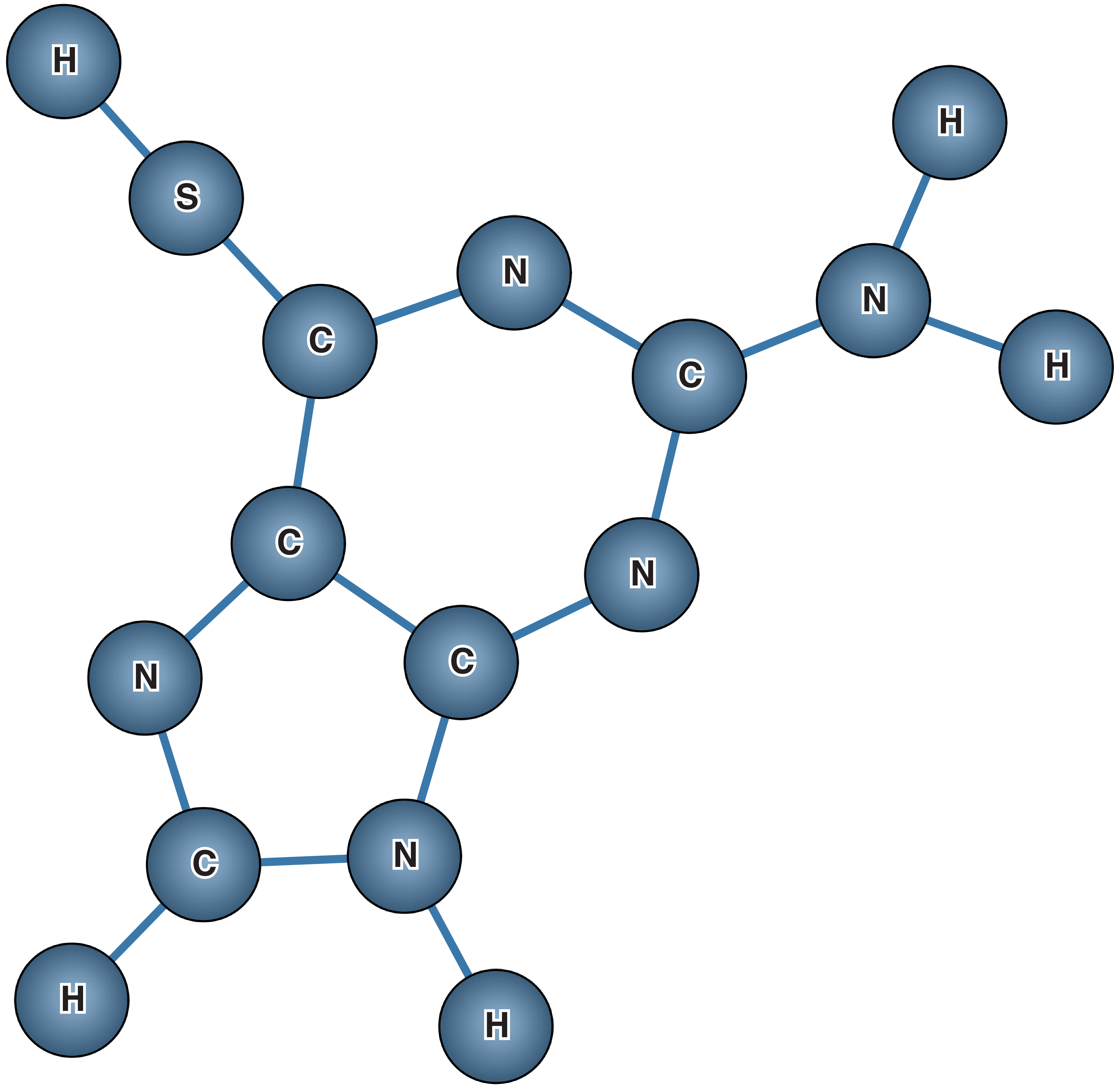
- Why not graphs?
  - Bar graph
  - Graphing functions in mathematics
- Network: nodes and edges connecting the nodes
- Formally, $G = (V,E)$ is a set of nodes V and a set of edges E where each edge connects two nodes.
- Nodes == items, edges connect items
- **Both** nodes and edges may have **attributes**

Node-Link Diagrams

- Data: nodes and edges
- Task: understand connectivity, paths, structure (topology)
- Encoding: nodes as point marks, connections as line marks
- Scalability: hundreds

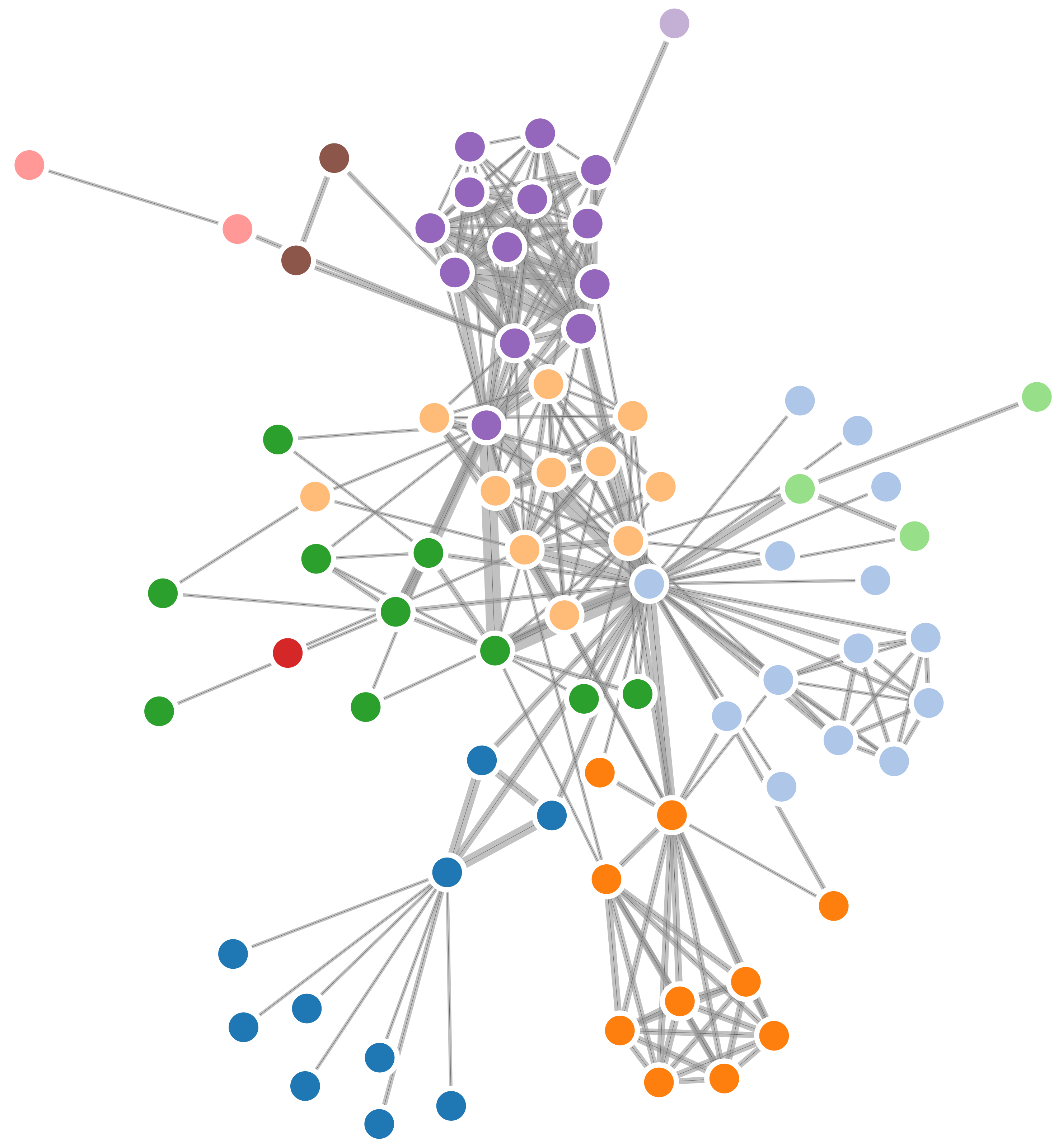
- …but high **density** of links can be problematic!

- We need position info—a **layout**!



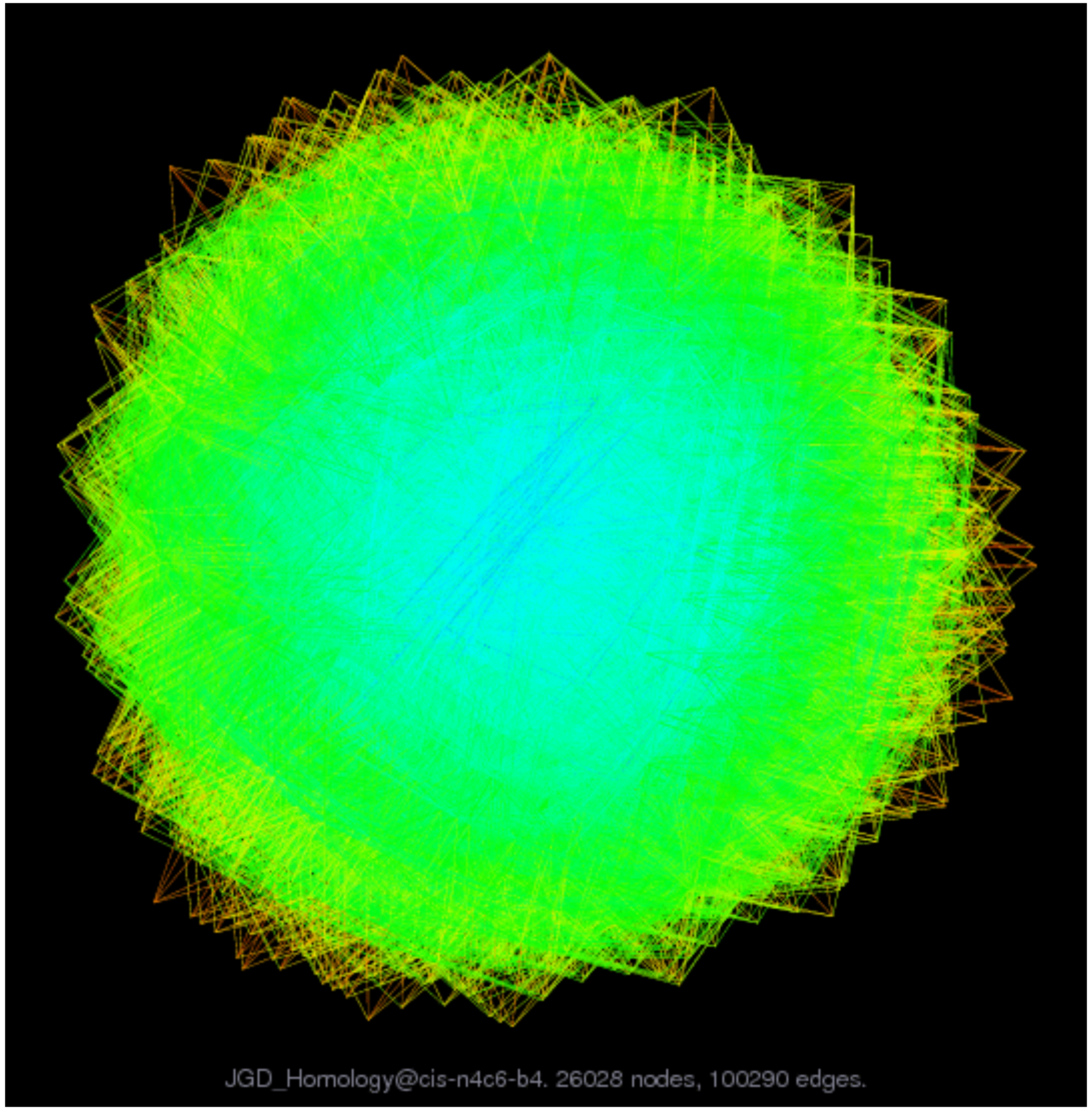
Force-Directed Layout

- Nodes push away from each other but edges are springs that pull them together
- Weakness: nondeterminism, algorithm may produce difference results each time it runs



[M. Bostock, 2017]

“Hairball”



[Hu, 2014]

Midterm

- Thursday
- Covers material through this week
- Format:
  - Multiple Choice
  - Free Response (often multi-part)
  - CS 680 students will have extra questions related to the research papers discussed

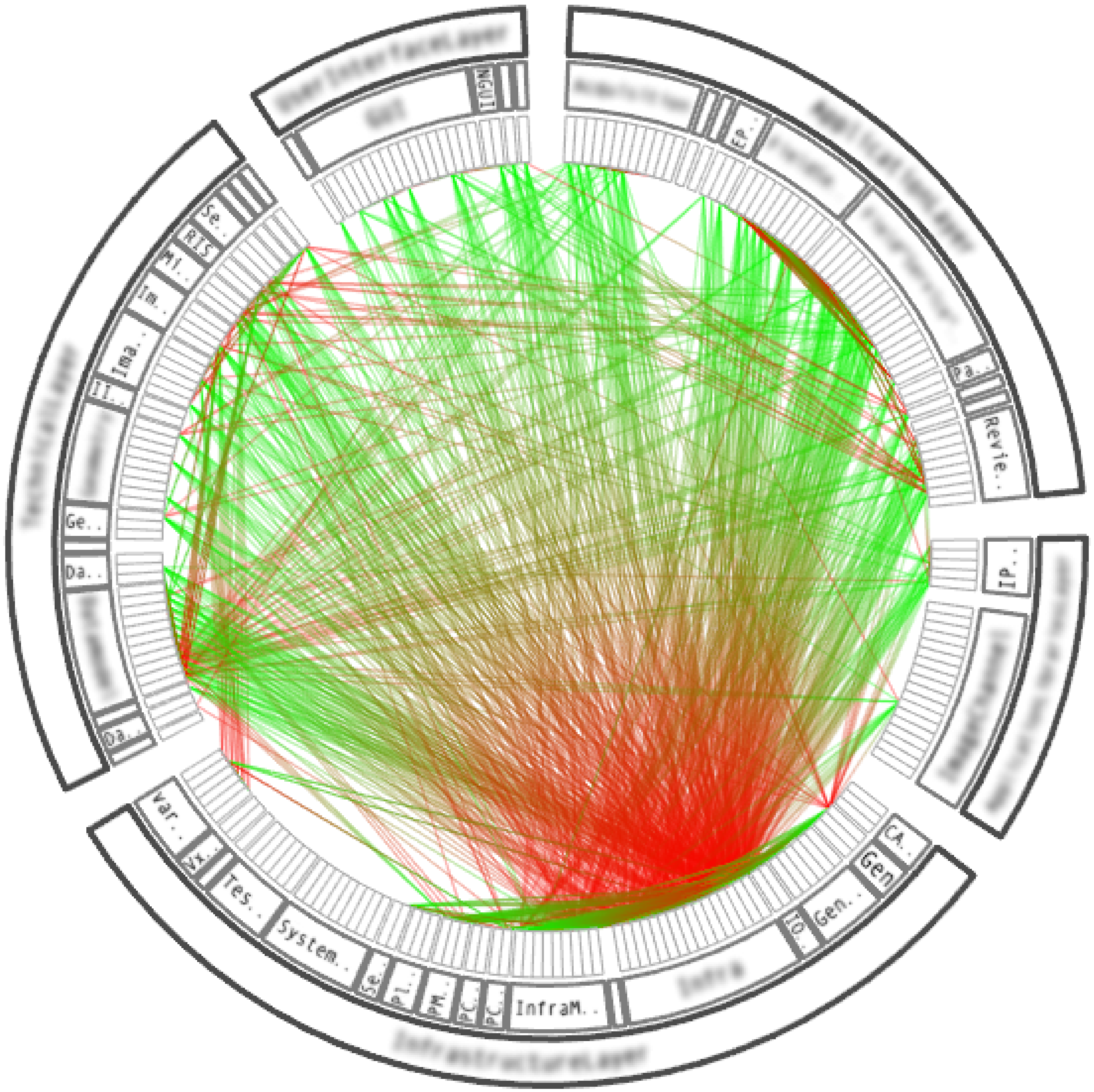
Project Design

- Start working on turning your visualization ideas into designs
- Sketch
- Options:
    - Try vastly different options
    - Refine an initial idea

Next Week

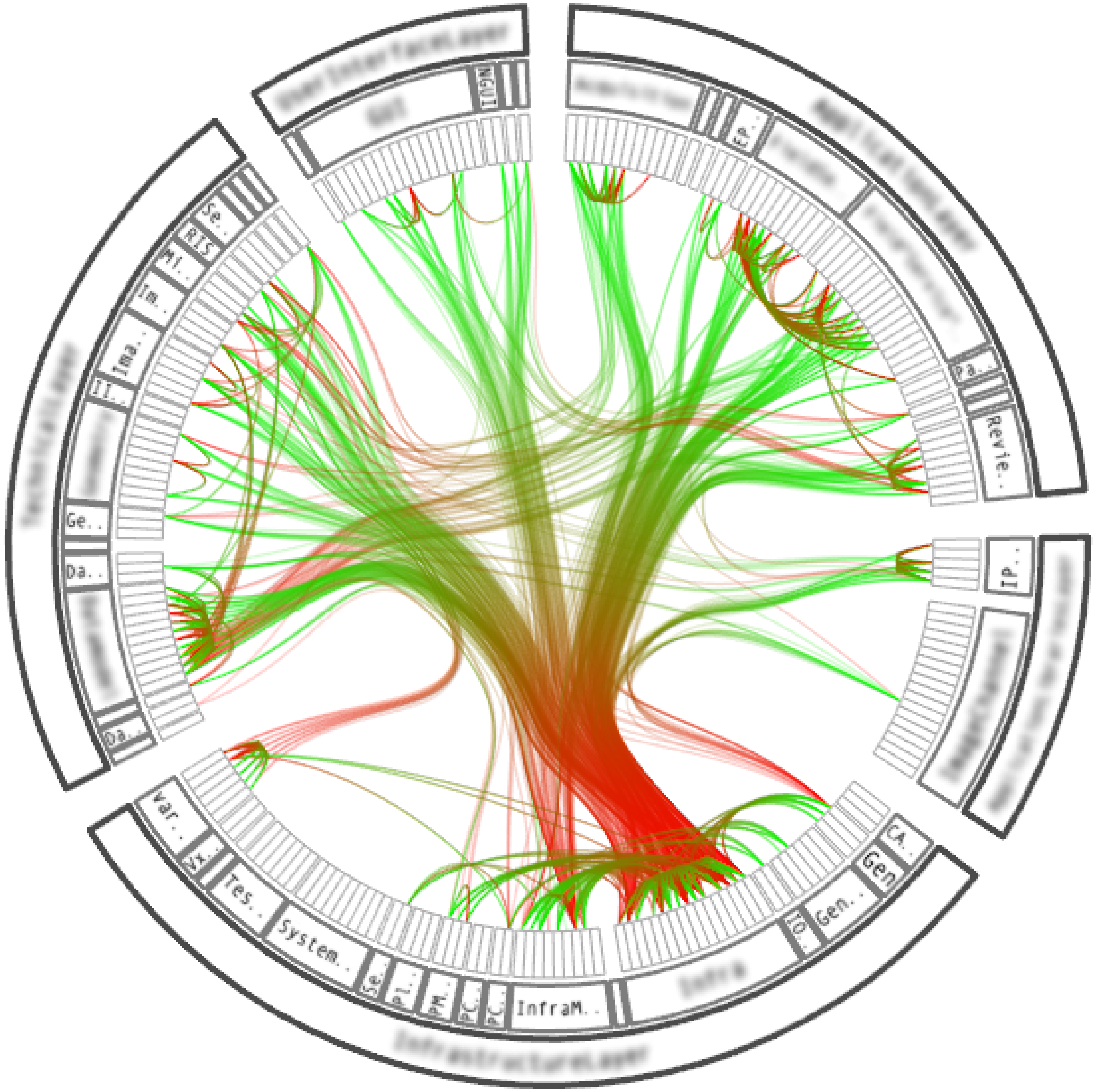
- Tuesday: Guest Lecture from Prof. Sun
- Thursday: No class, work on projects

Hierarchical Edge Bundling



[Holten, 2006]

Hierarchical Edge Bundling



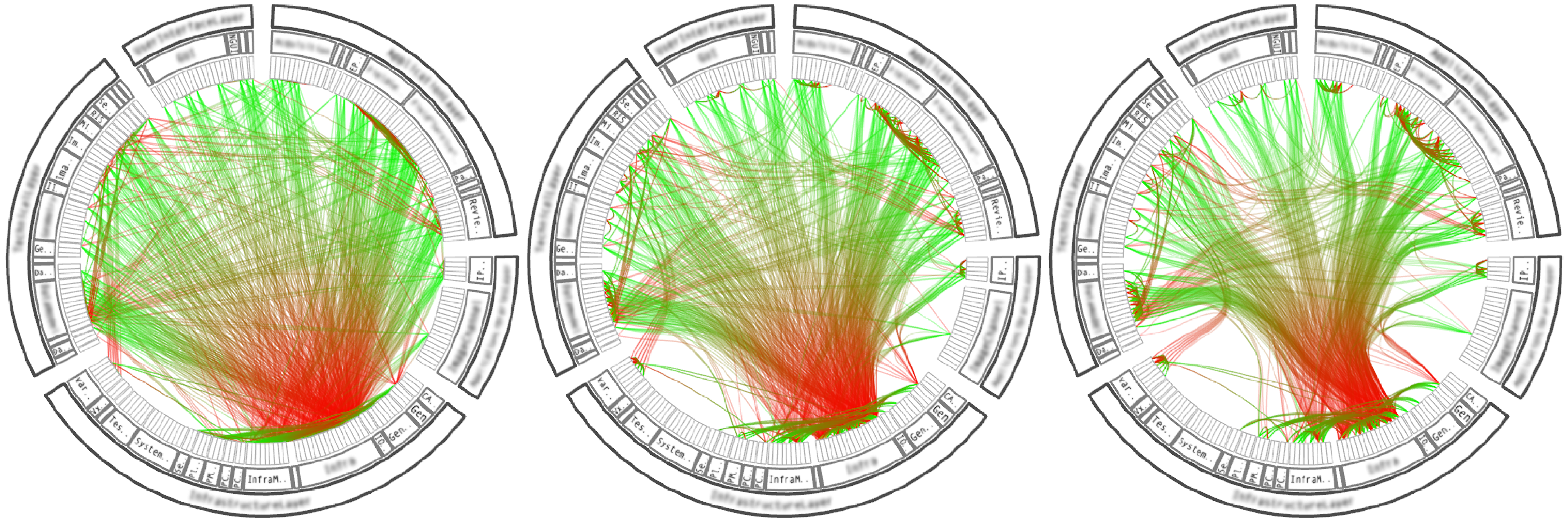
[Holten, 2006]

Hierarchical Edge Bundling

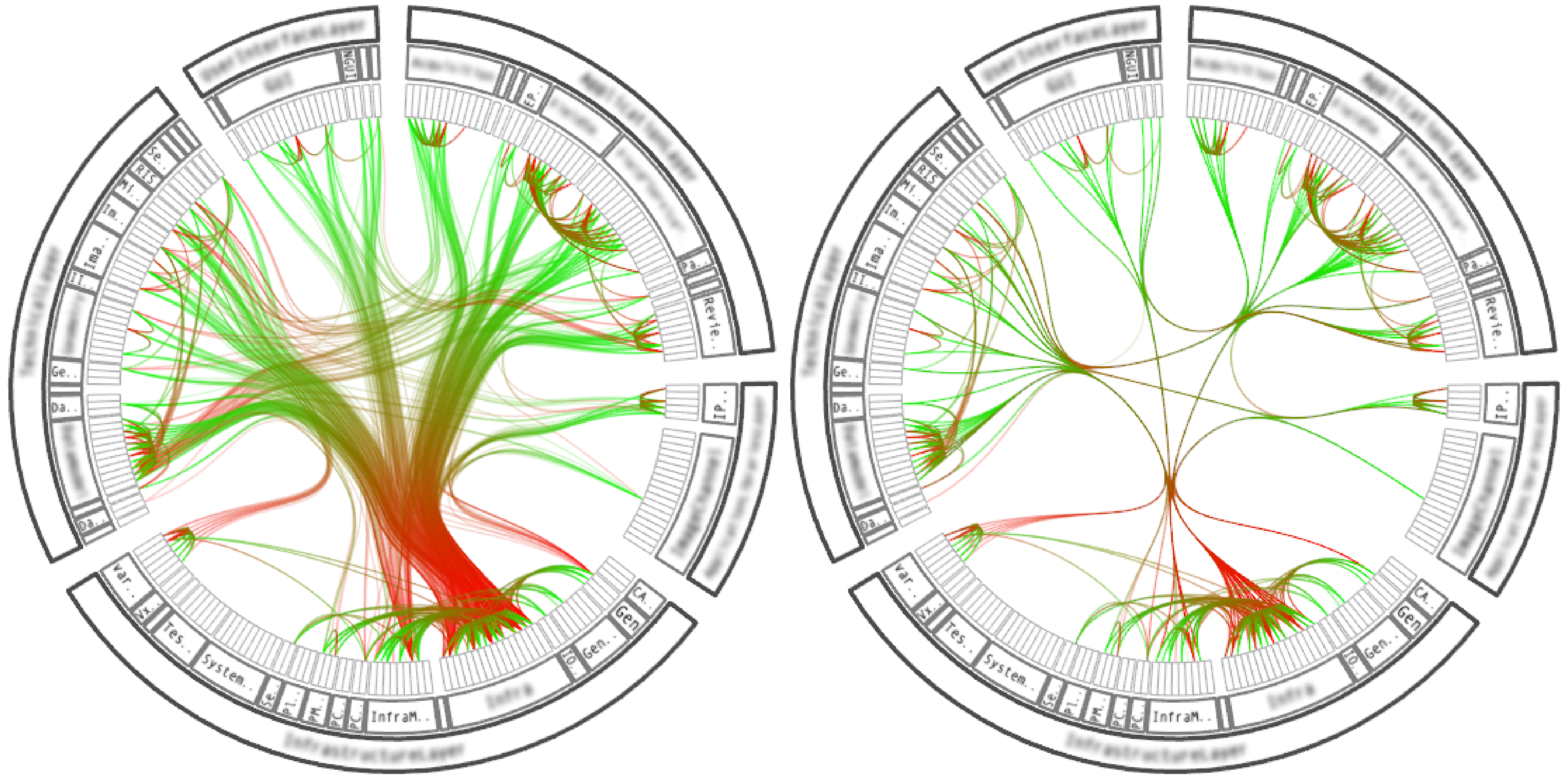
- Flexible and generic method
- Reduces visual clutter when dealing with large numbers of adjacency edges
- Provides an intuitive and continuous way to control the strength of bundling.
  - Low bundling strength mainly provides low-level, node-to-node connectivity information
  - High bundling strength provides high-level information as well by implicit visualization of adjacency edges between parent nodes that are the result of explicit adjacency edges between their respective child nodes

[Holten, 2006]

Bundling Strength



$\beta = 0$ $\beta = 0.25$ $\beta = 0.5$



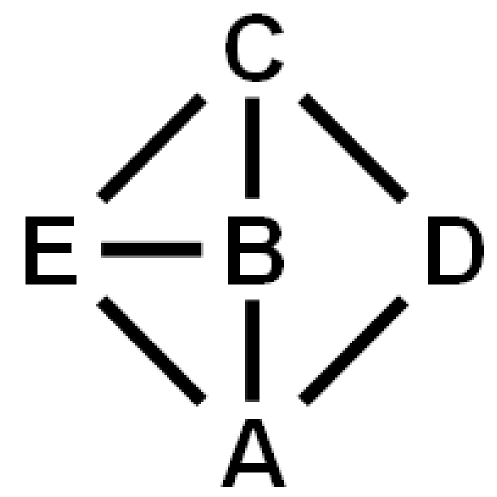
$\beta = 0.75$ $\beta = 1$

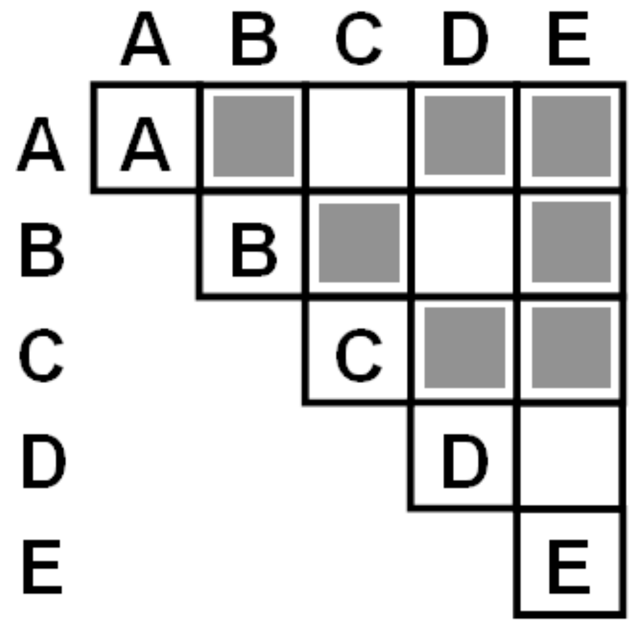
[Holten, 2006]

Adjacency Matrix

- Change network to tabular data and use a matrix representation
- Derived data: nodes are keys, edges are boolean values
- Task: lookup connections, find well-connected clusters
- Scalability: millions of edges

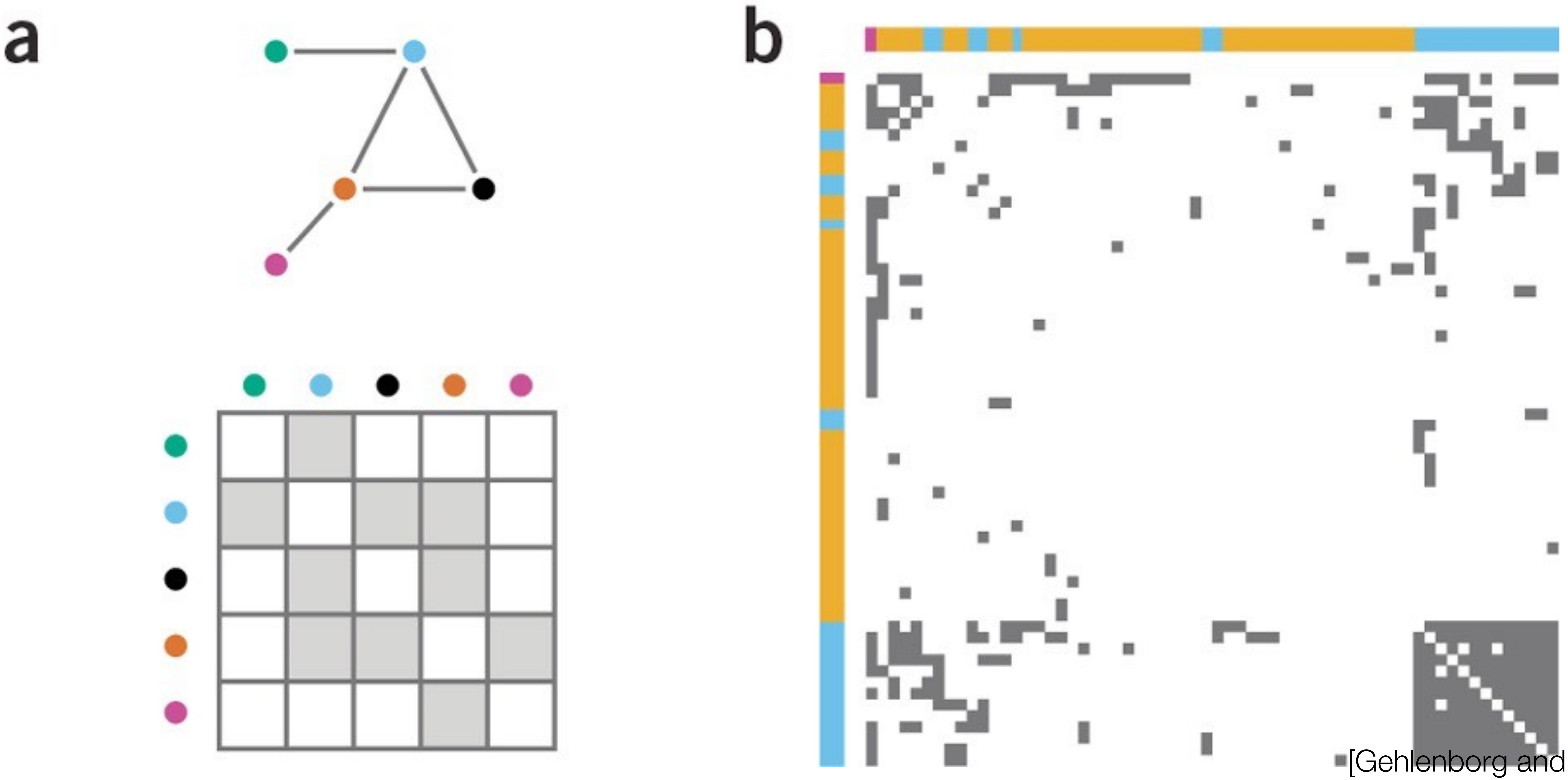
- Can encode **edge weight**, too





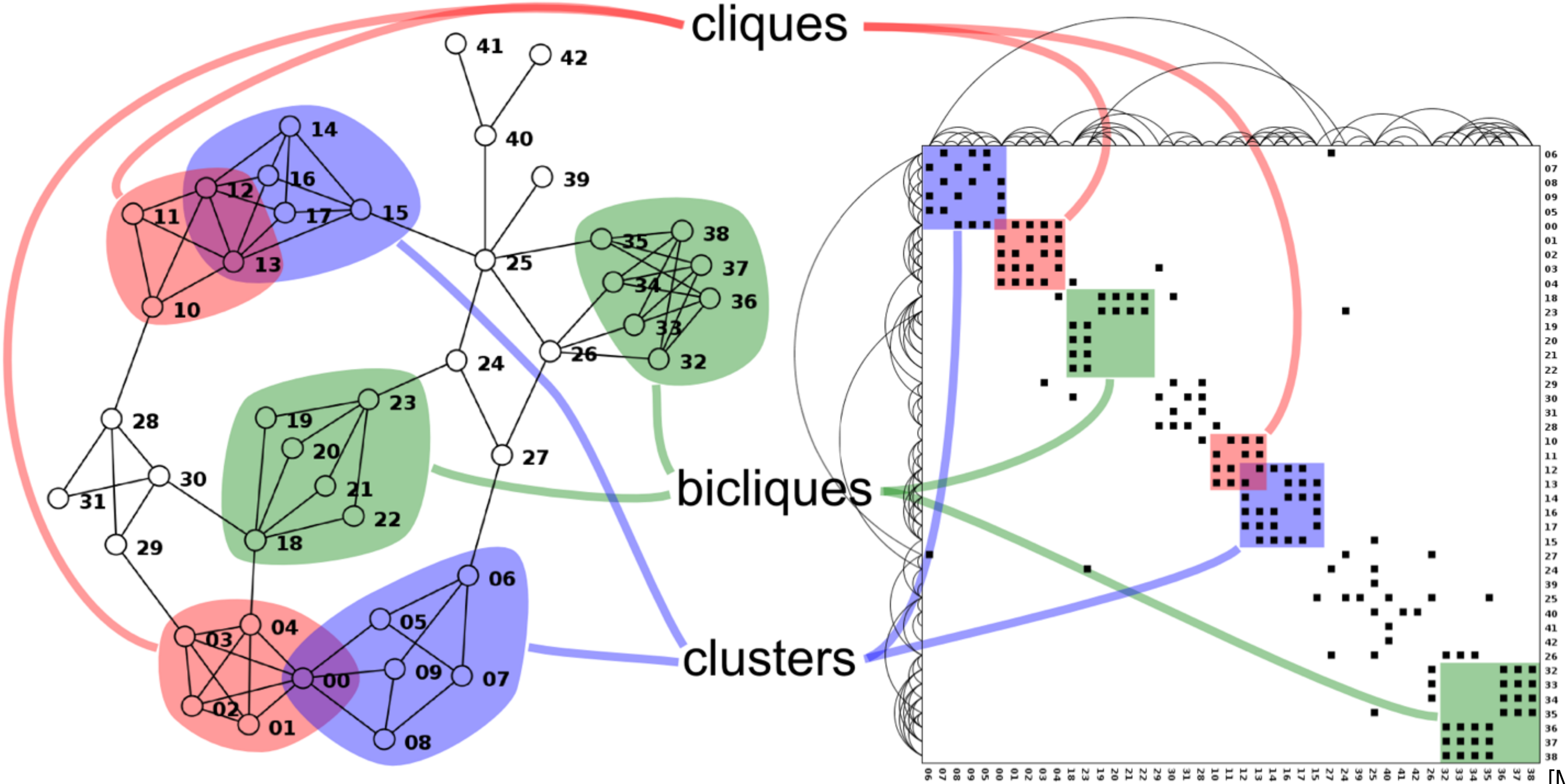
[Henry et al., 2007]

Cliques in Adjacency Matrices



[Gehlenborg and Wong]

Structures from Adjacency Matrices



[McGuffin]

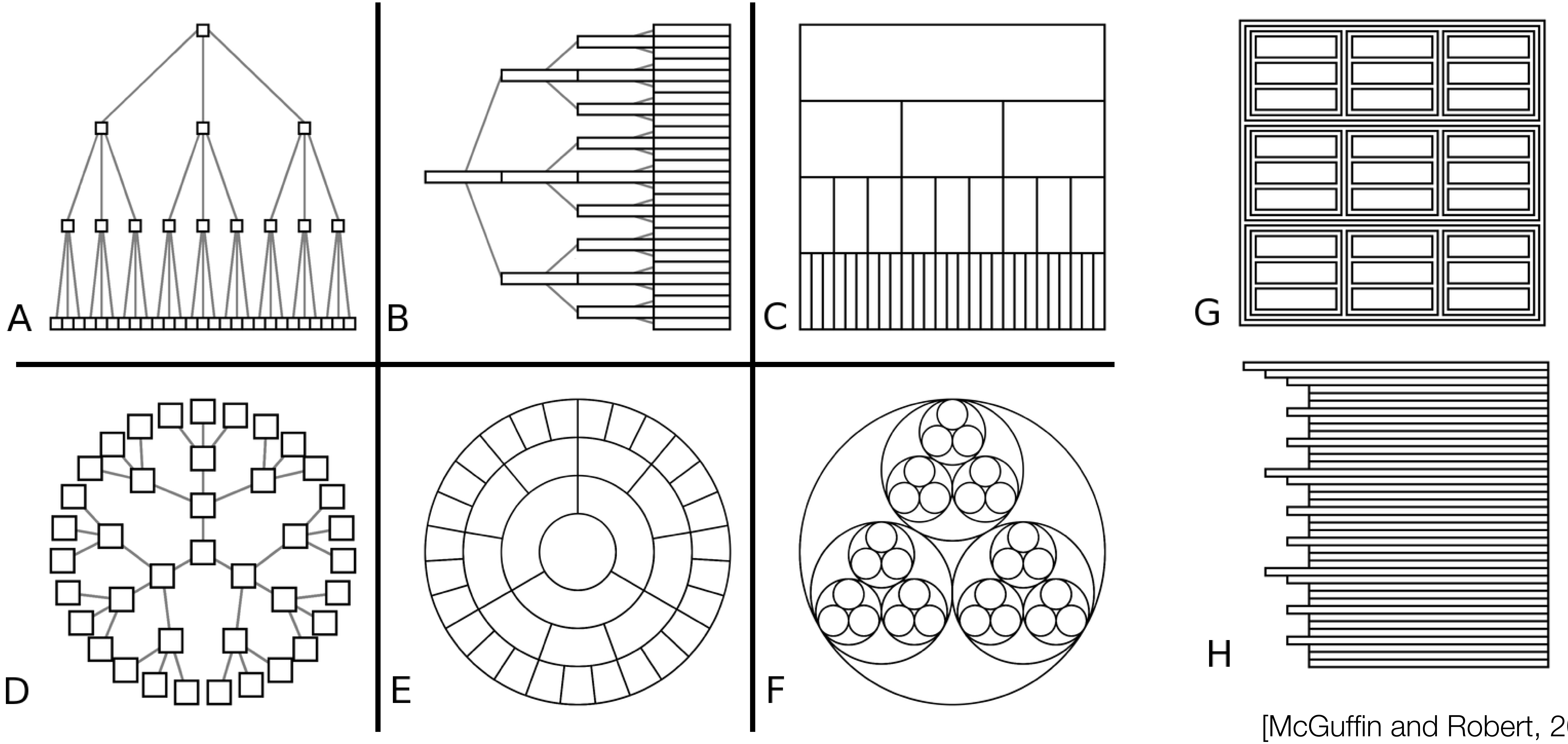
Node-Link or Adjacency Matrix?

- Empirical study: For most tasks, node-link is better for small graphs and adjacency better for large graphs
- Multi-link paths are hard with adjacency matrices
- Immediate connectivity or neighbors are ok, estimating size (nodes & edges also ok)
- People tend to be more familiar with node-link diagrams
- Link density is a problem with node-link but not with adjacency matrices

Trees

- Trees are directed acyclic **graphs**
  - each edge has a direction: the origin is the parent, the destination is the child
  - cannot get back to a node after leaving it
- …plus each node has **at most one** parent node
- A tree has a **root** (every other node hangs off it)
- Can consider enclosure in trees using parent-child relationships

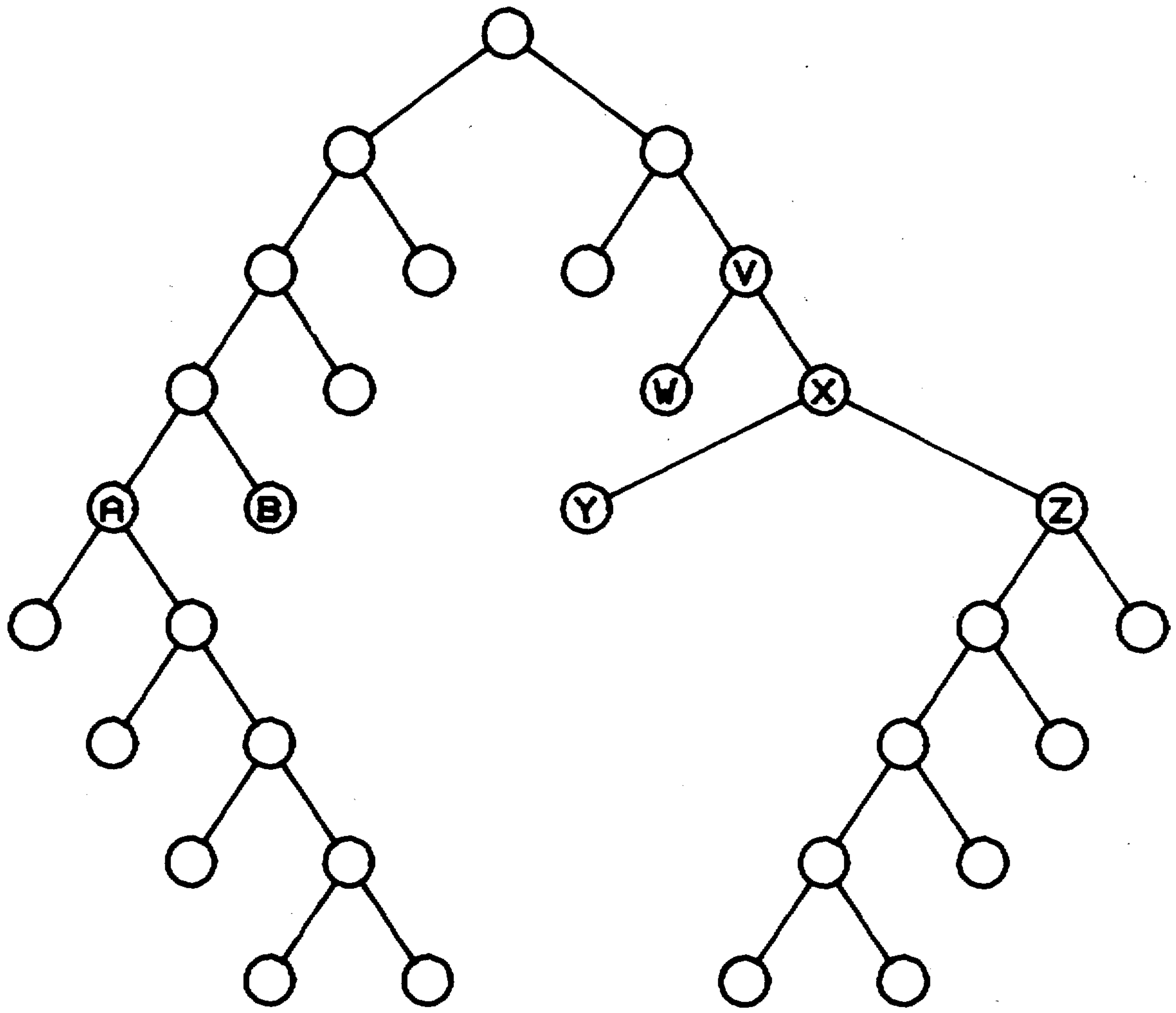
Tree Visualizations



[McGuffin and Robert, 2010]

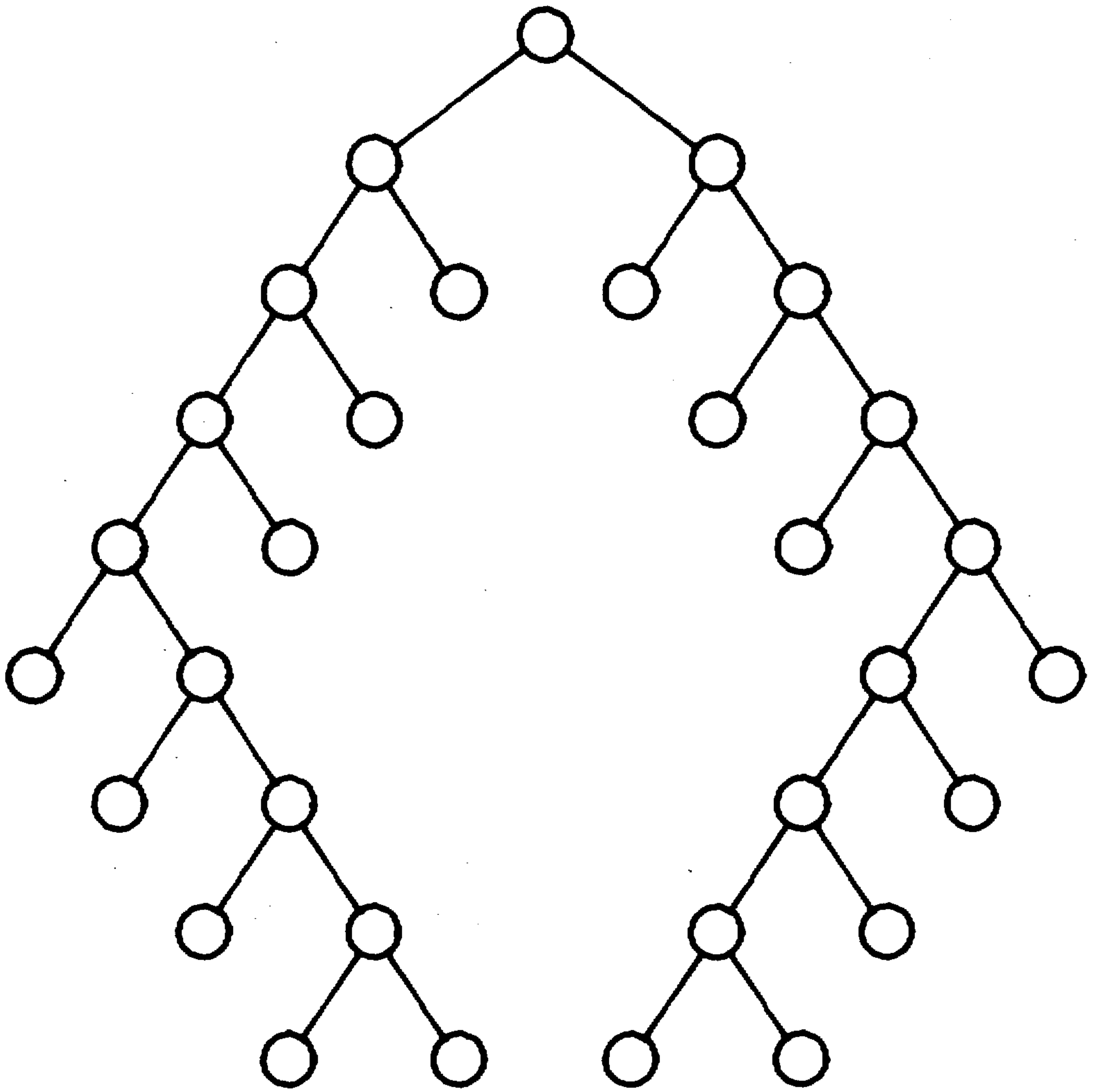
Node-Link Diagram

- Trees are graphs
- …but we have more structure
- Horizontal or vertical
- Idea 1: partition space for each node via recursion
- Idea 2: “Tidy” Drawing
  - Wetherell & Shannon: Don’t waste space (overlapping parent nodes is ok)
  - Reingold and Tilford: Keep symmetry, subtrees look similar



[WS Alg., Reingold and Tilford, 1981]

Reingold-Tilford Algorithm

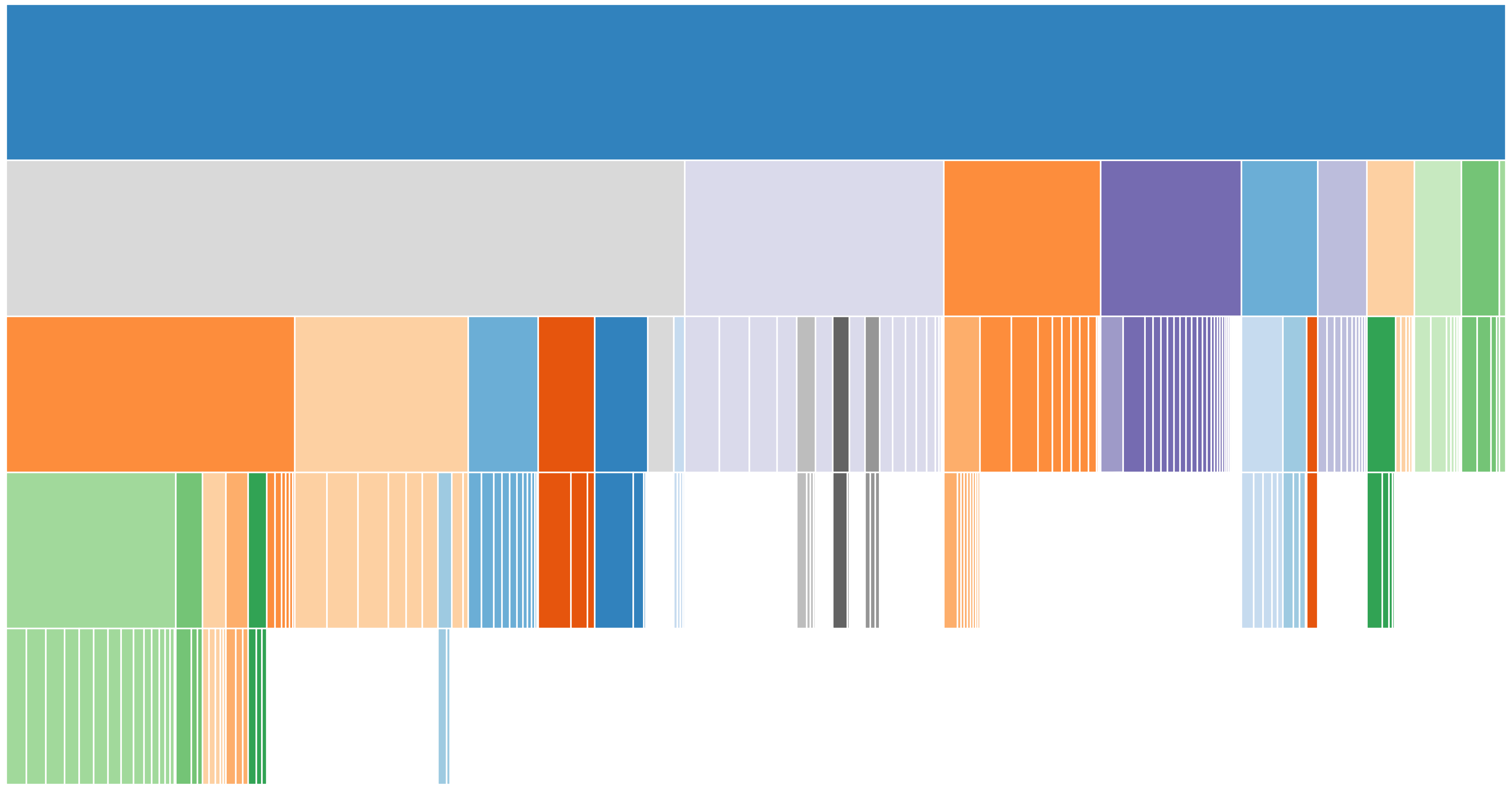


- Recurse on left and right subtrees
- Shift subtree over as long as it doesn't overlap
- Place parent centered above the subtrees
- Originally, only binary trees, extended by Walker

[Reingold and Tilford, 1981]

Icicle Plot

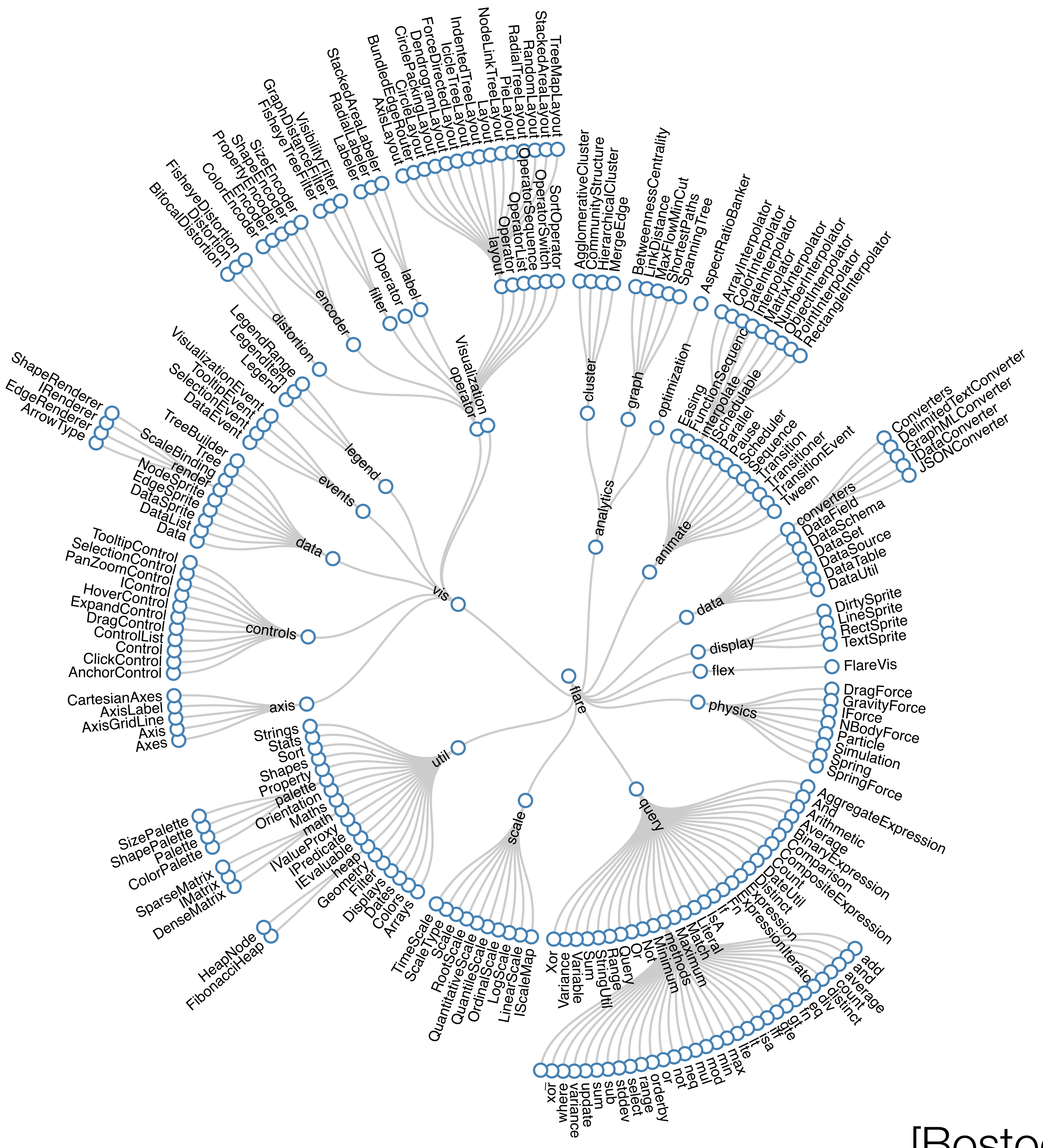
- Line marks
- Vertical position shows depth
- Horizontal position shows links and sibling order
- Scalability: 1 pixel leaves, but harder to label



[Bostock, 2011]

Radial Node-Link

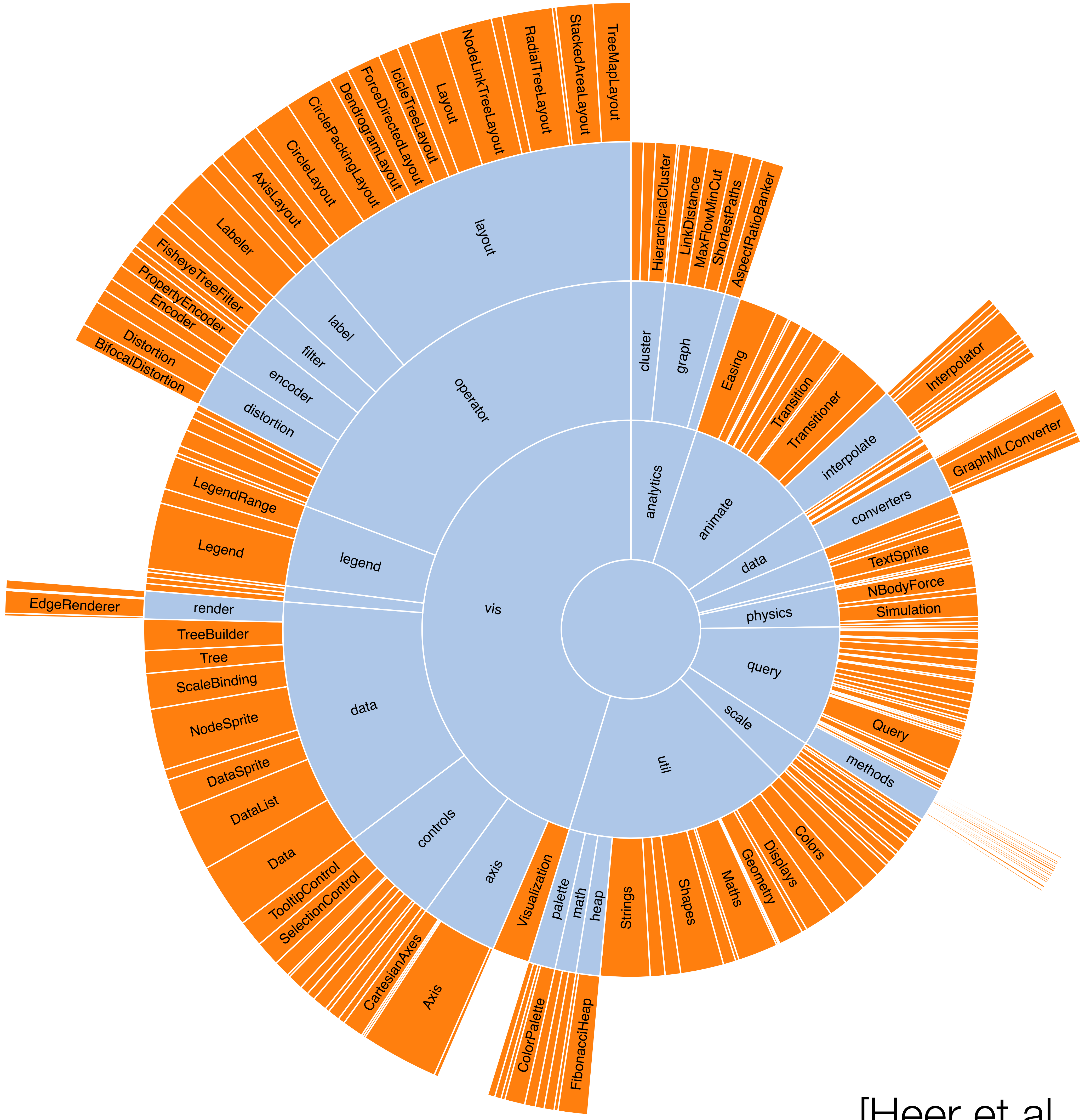
- Use polar coordinates instead of rectilinear
- Same layout algorithms work (e.g. Reingold-Tilford)
- Benefit: space usage, labels



[Bostock, 2012]

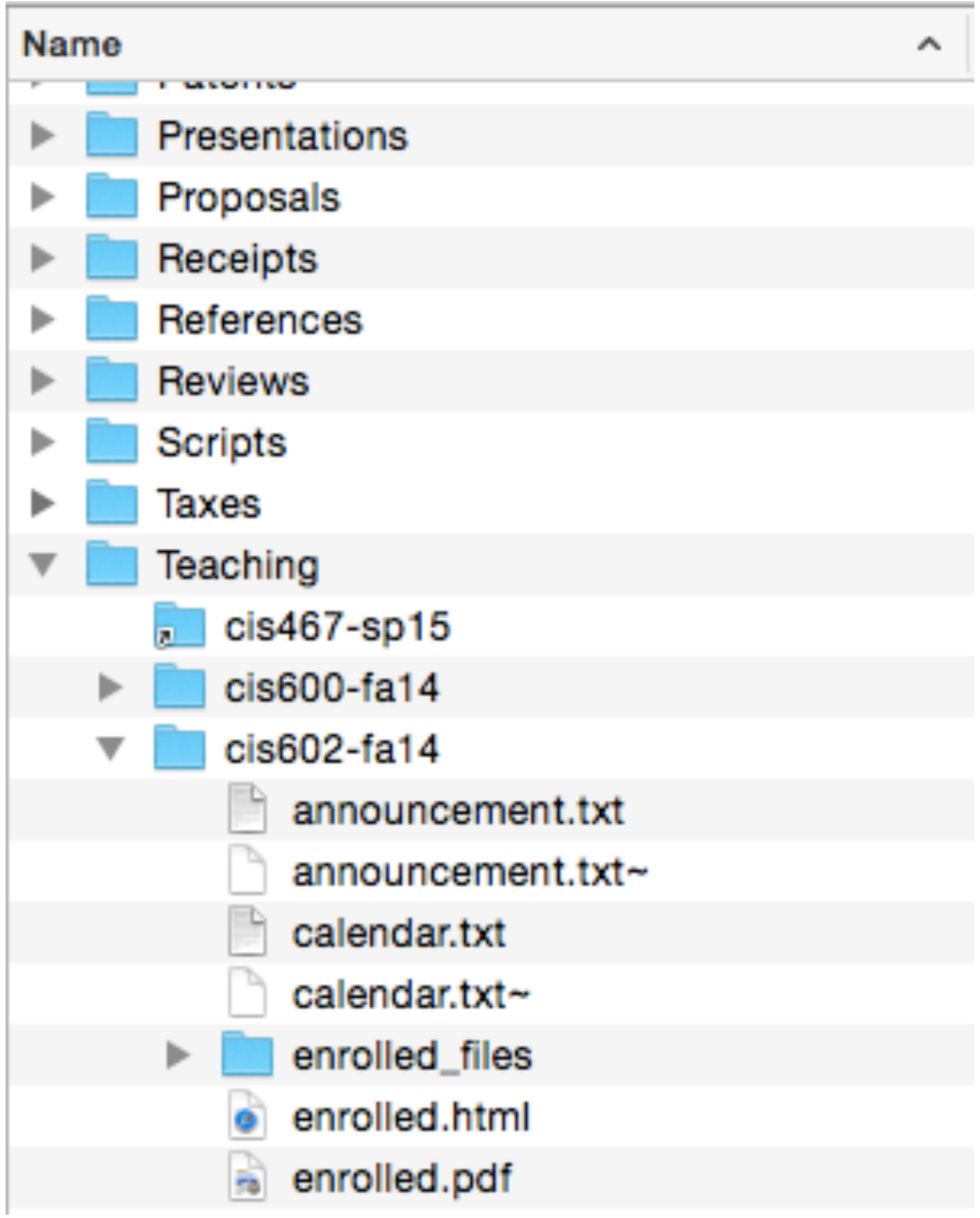
Sunburst

- Icicle plot in a radial layout
- Reading labels?
- Intuitive navigation



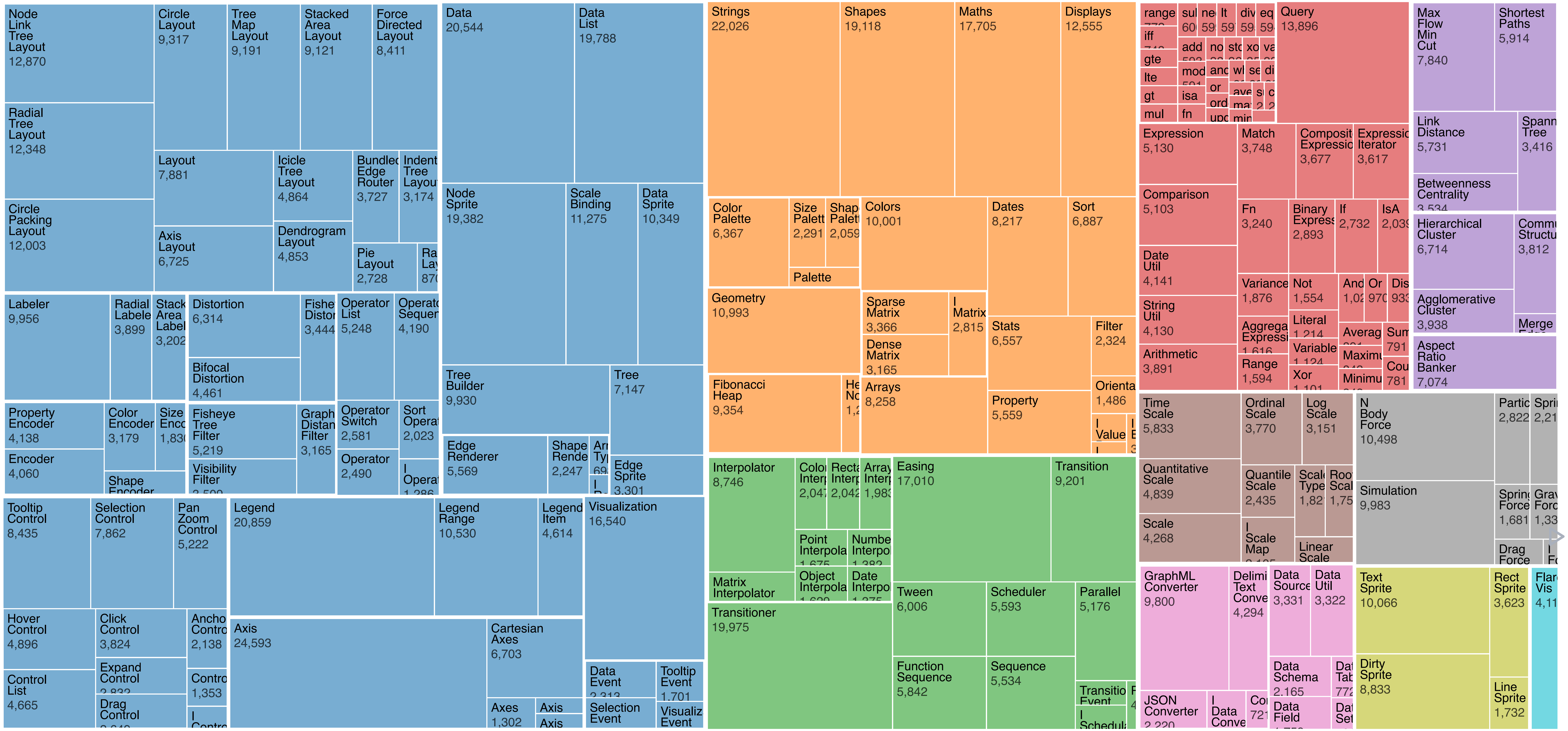
[Heer et al., 2012]

Indented Outline



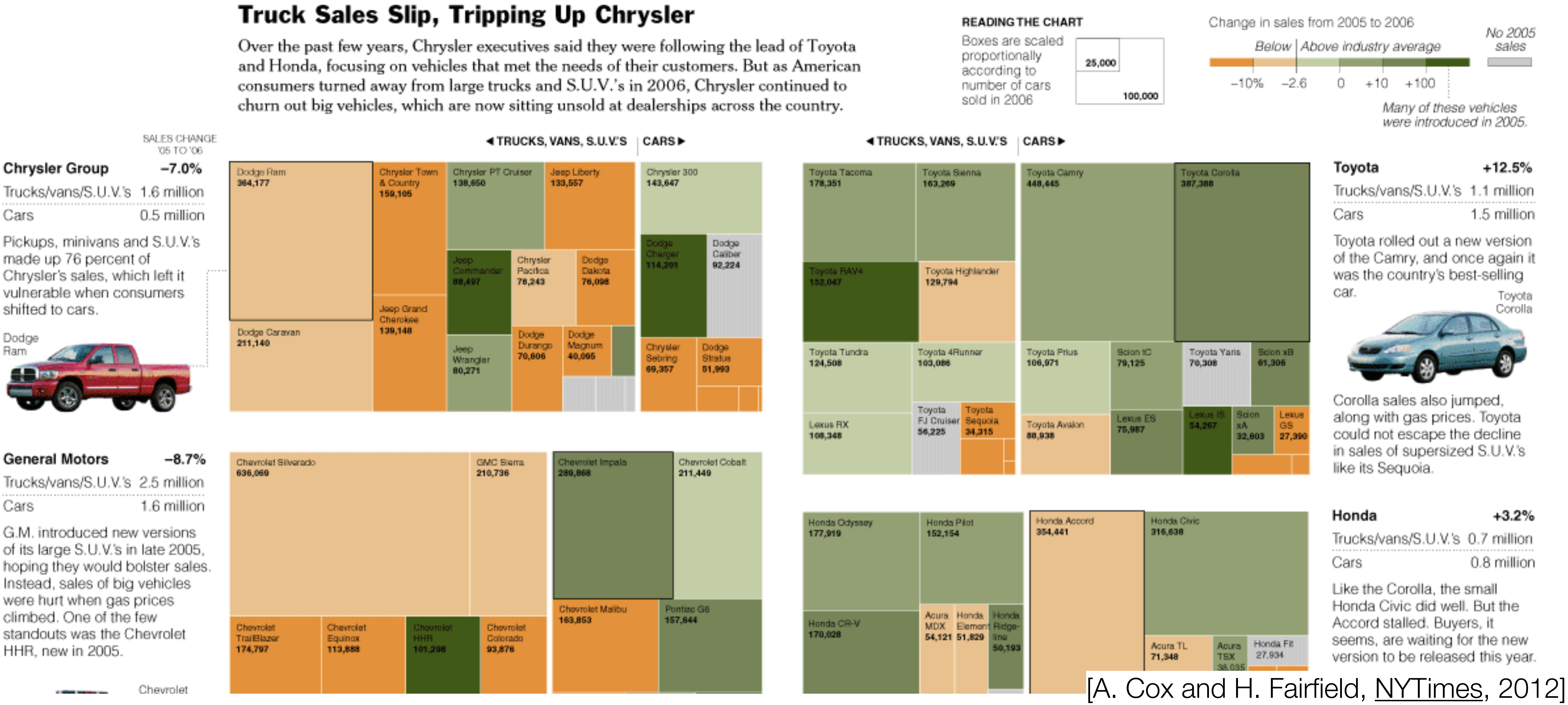
- Like a filesystem tree
- Use horizontal position to show depth, vertical positions show sibling/order

Treemap



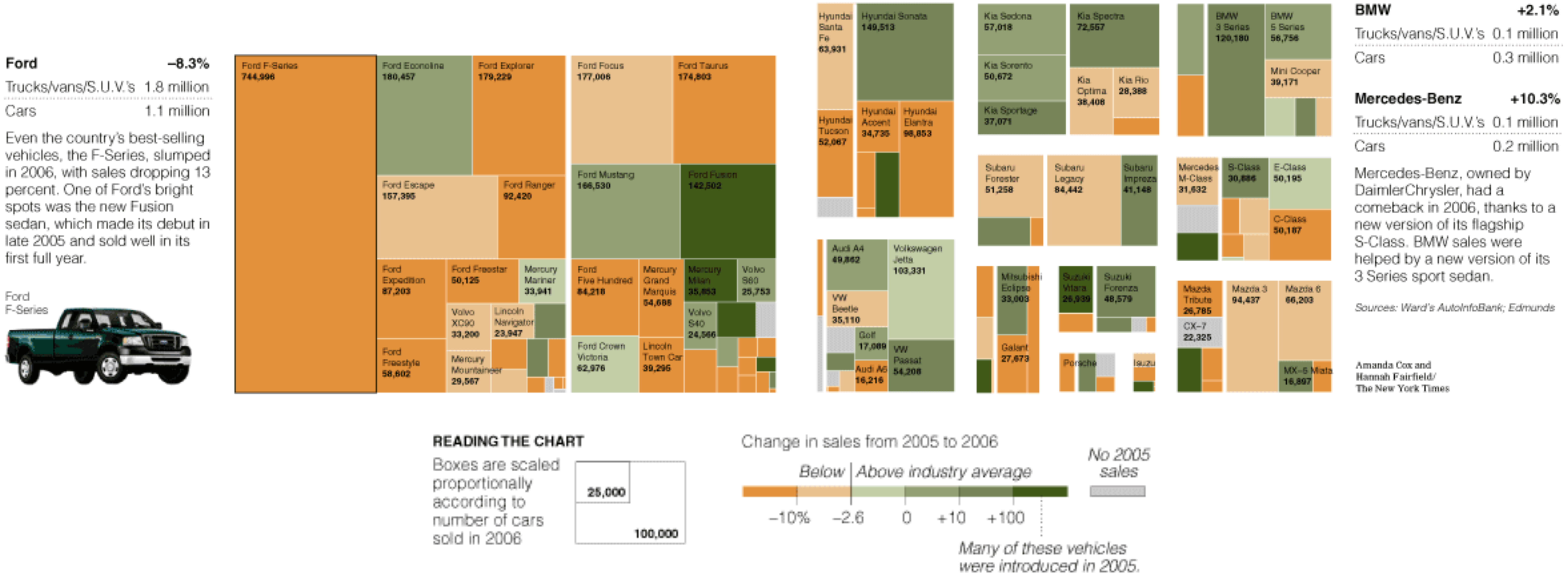
[M. Bostock, 2017]

Car/Truck Treemap



[A. Cox and H. Fairfield, NYTimes, 2012]

Car/Truck Treemap



[A. Cox and H. Fairfield, NYTimes, 2012]

Treemap

- Containment marks instead of connection marks
- Encodes some attribute of the items as the **size** of the rectangles
- Not as easy to see the intermediate rectangles
- Scalability: millions of leaf nodes and links possible

- Need a layout algorithm!

Layout Algorithms

- How do we generate the area marks?
- What considerations should we try to keep in mind?

Layout Algorithms

- How do we generate the area marks?
- What considerations should we try to keep in mind?
  - area true to quantitative value
  - show hierarchy
  - aspect ratio
- Also…
  - ordering
  - stability

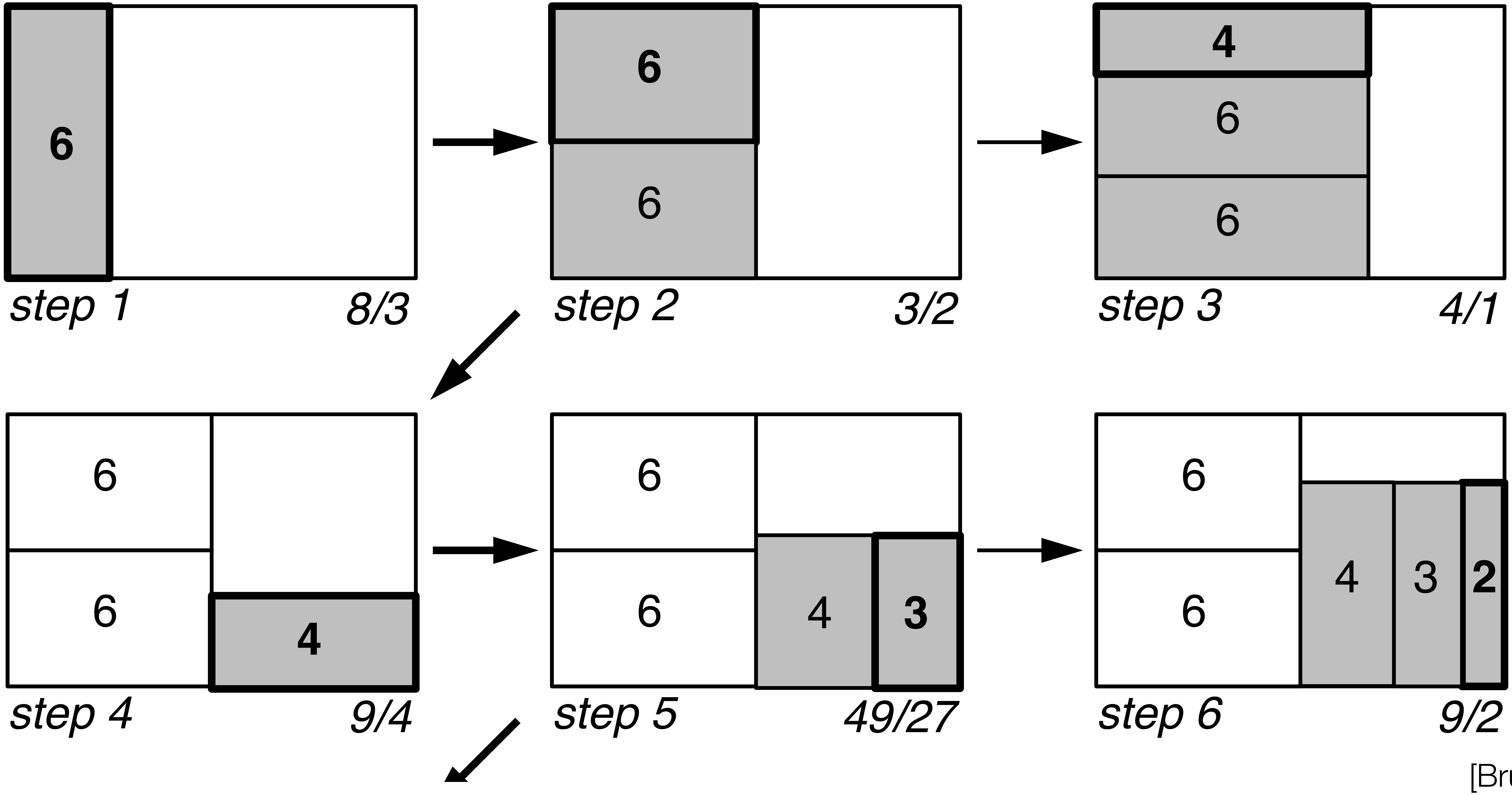
Layout Algorithms

- **Aspect ratio** concerns: square is better
- Slice and dice:
  - Split at each level into strips
  - At each step, orientation of division (horiz/vert) changes
- Strip
  - Order rectangles and move to a new row when aspect ratio gets worse

Improving Treemaps (Squarified)

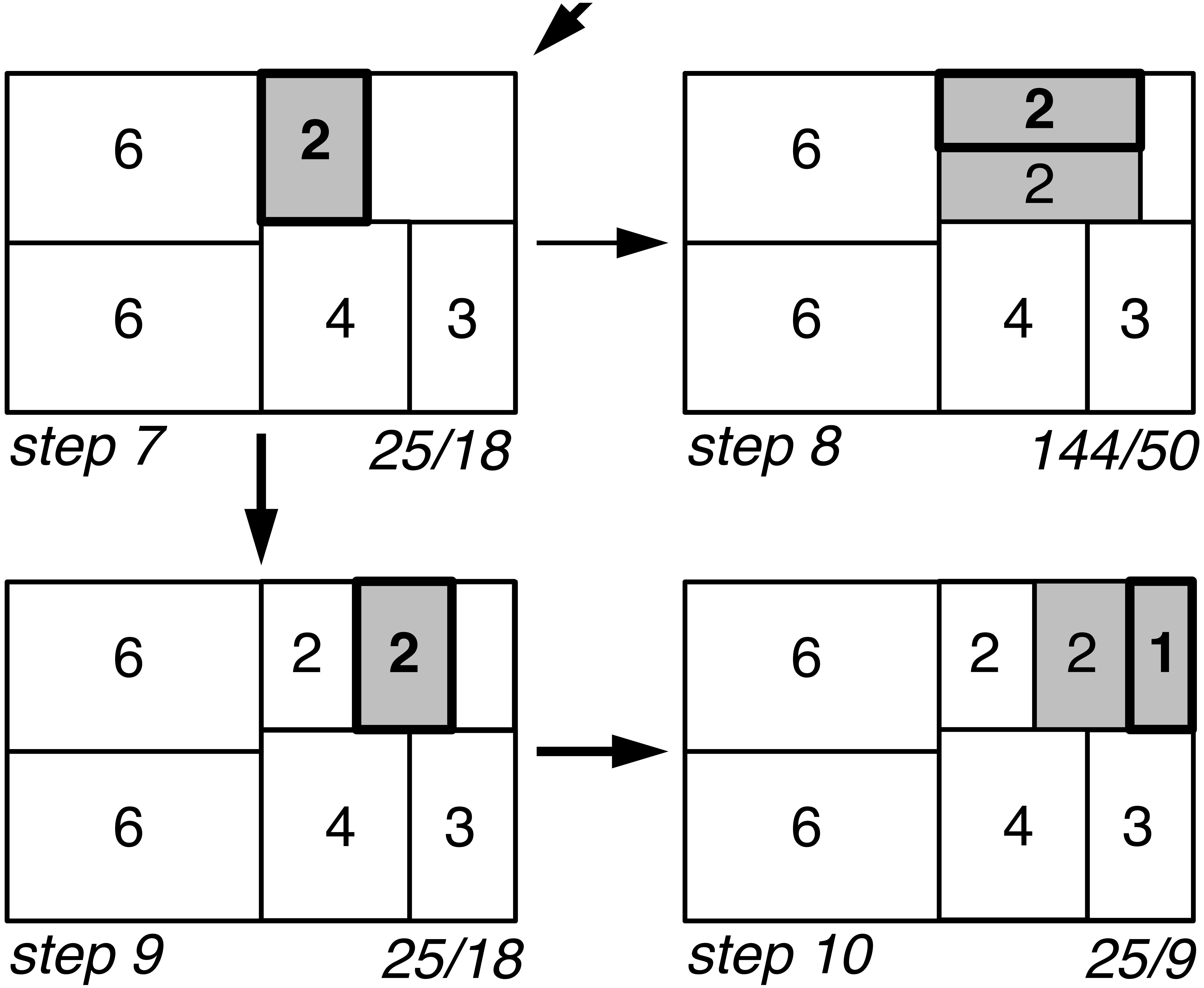
- Switching from horizontal to vertical cuts may be ok for nicely-behaved trees, but can lead to bad **aspect ratios**
- Problem: harder to compare sizes, more difficult to select/mouse over the rectangles
- Solution: Choose divisions (x/y) based on the width/height of region in order to maintain good aspect ratios
  - use left and right side
  - process large rectangles first
- Ordering not preserved which may cause issues if the data is updated

Squarification Algorithm



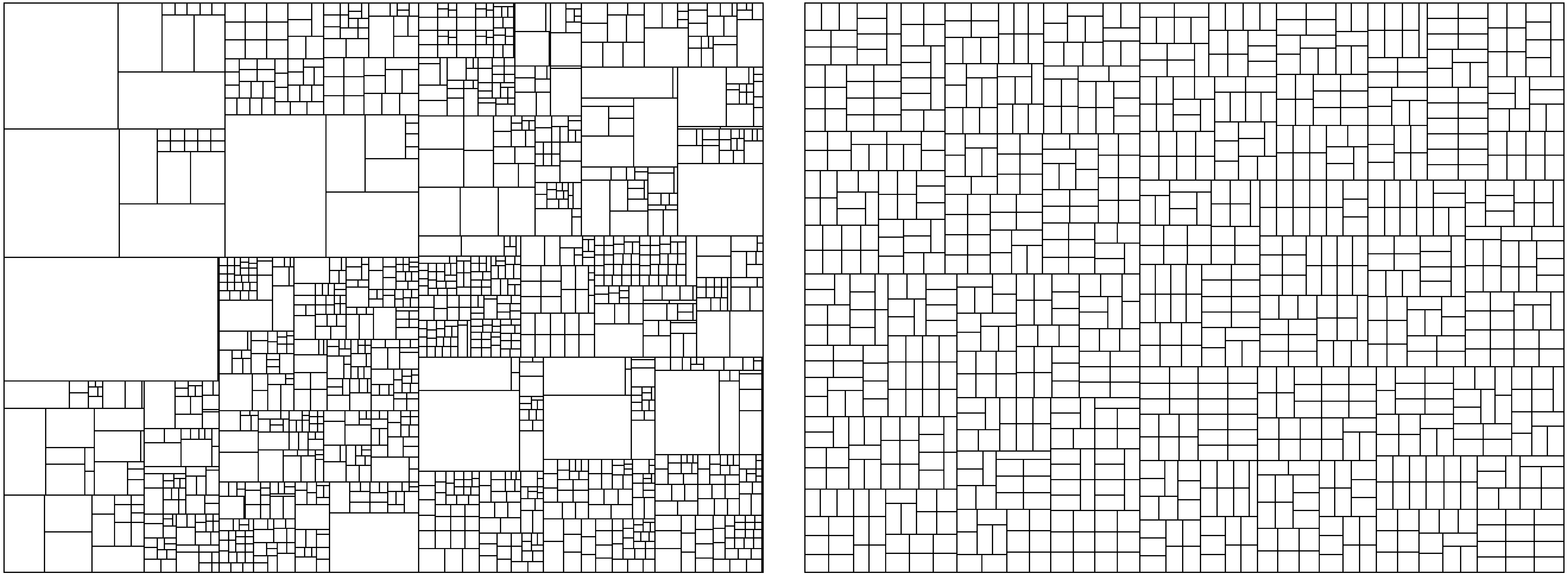
[Brus et al., 1999]

Squarification Algorithm



[Brus et al., 1999]

Squarified Treemaps



(a) File system (b) Organization

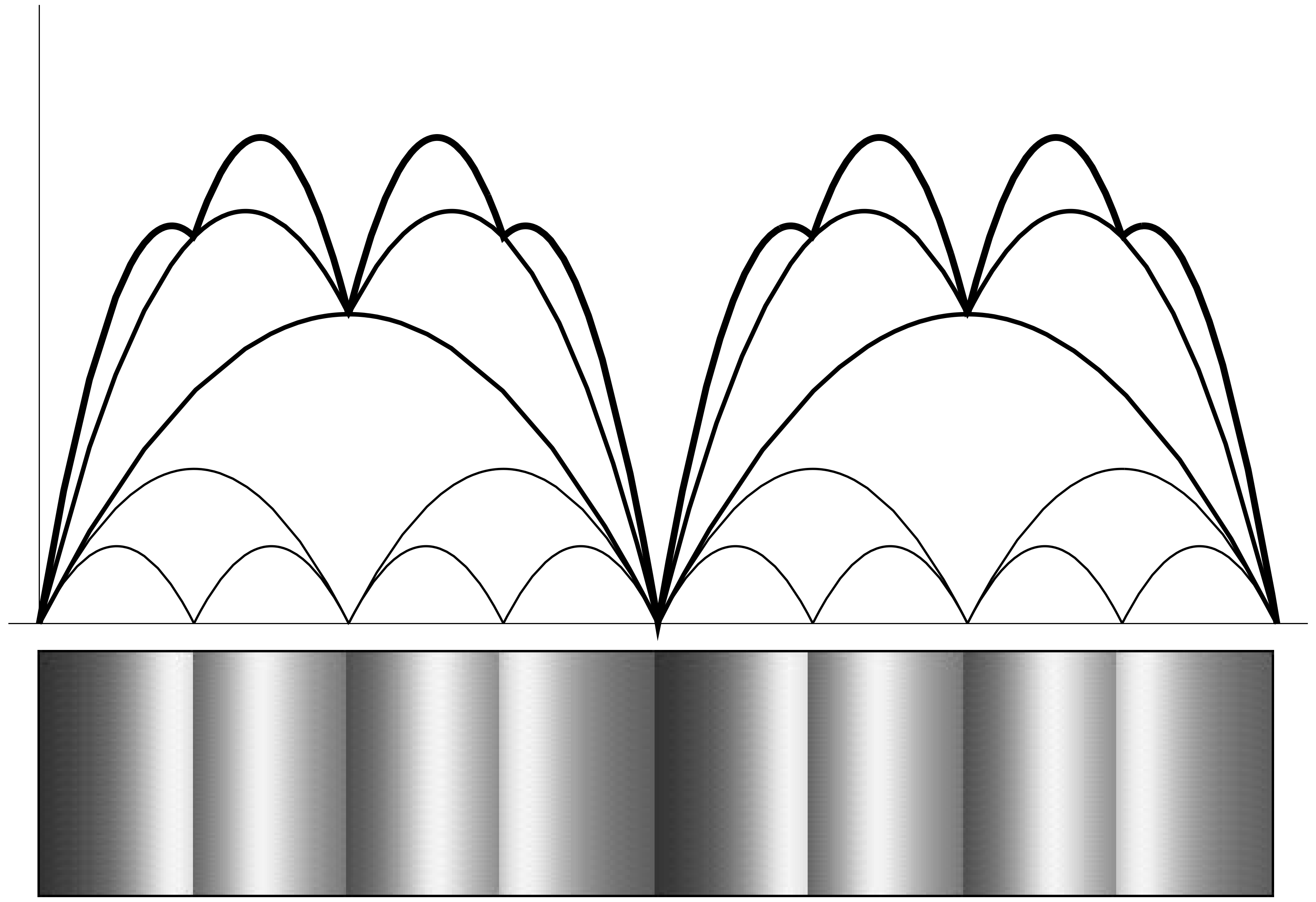
[Brus et al., 1999]

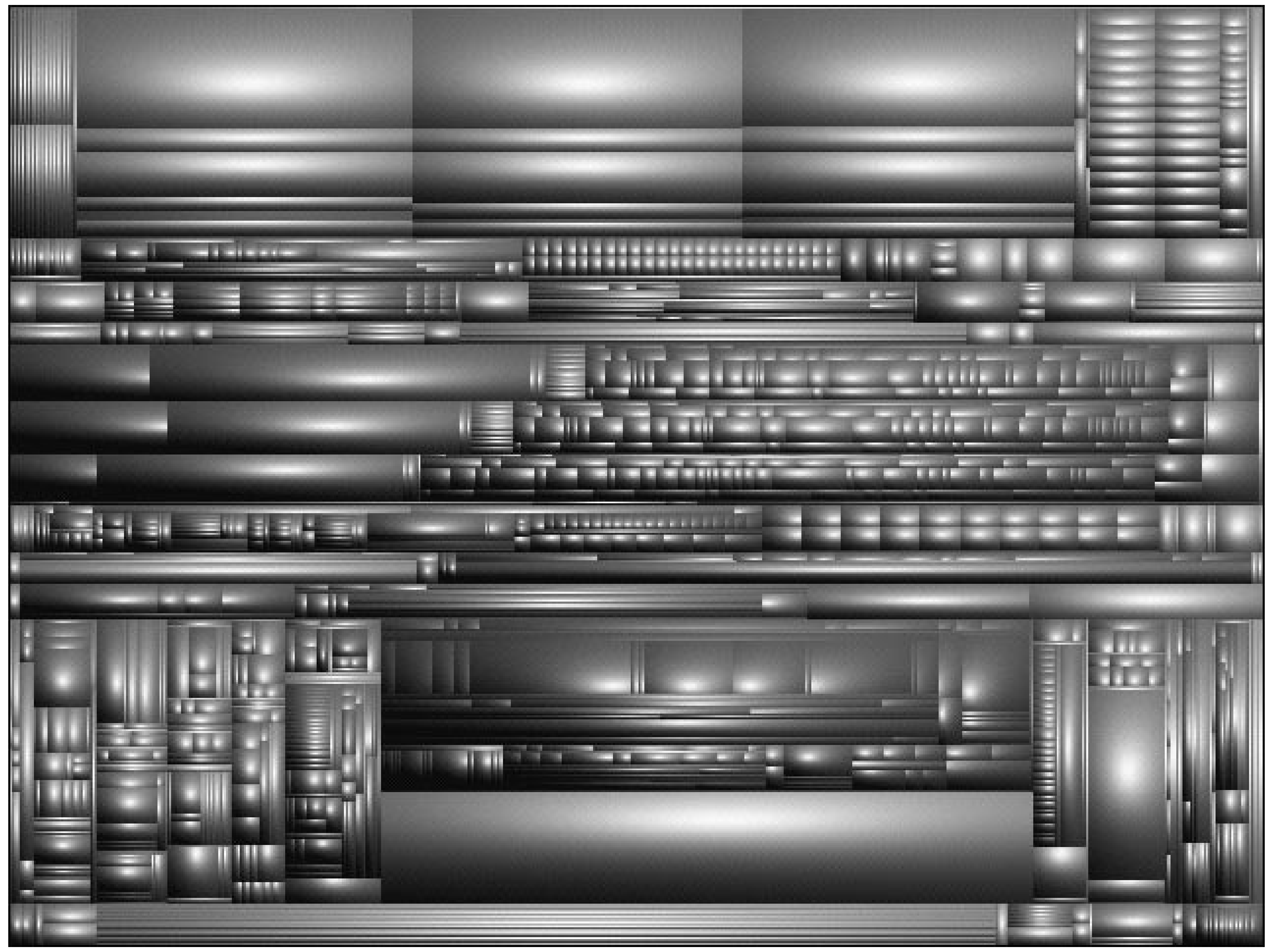
Squarified Layout

- Sort values
- Switch orientation whenever necessary to obtain best aspect ratios

Improving Treemaps (Cushion)

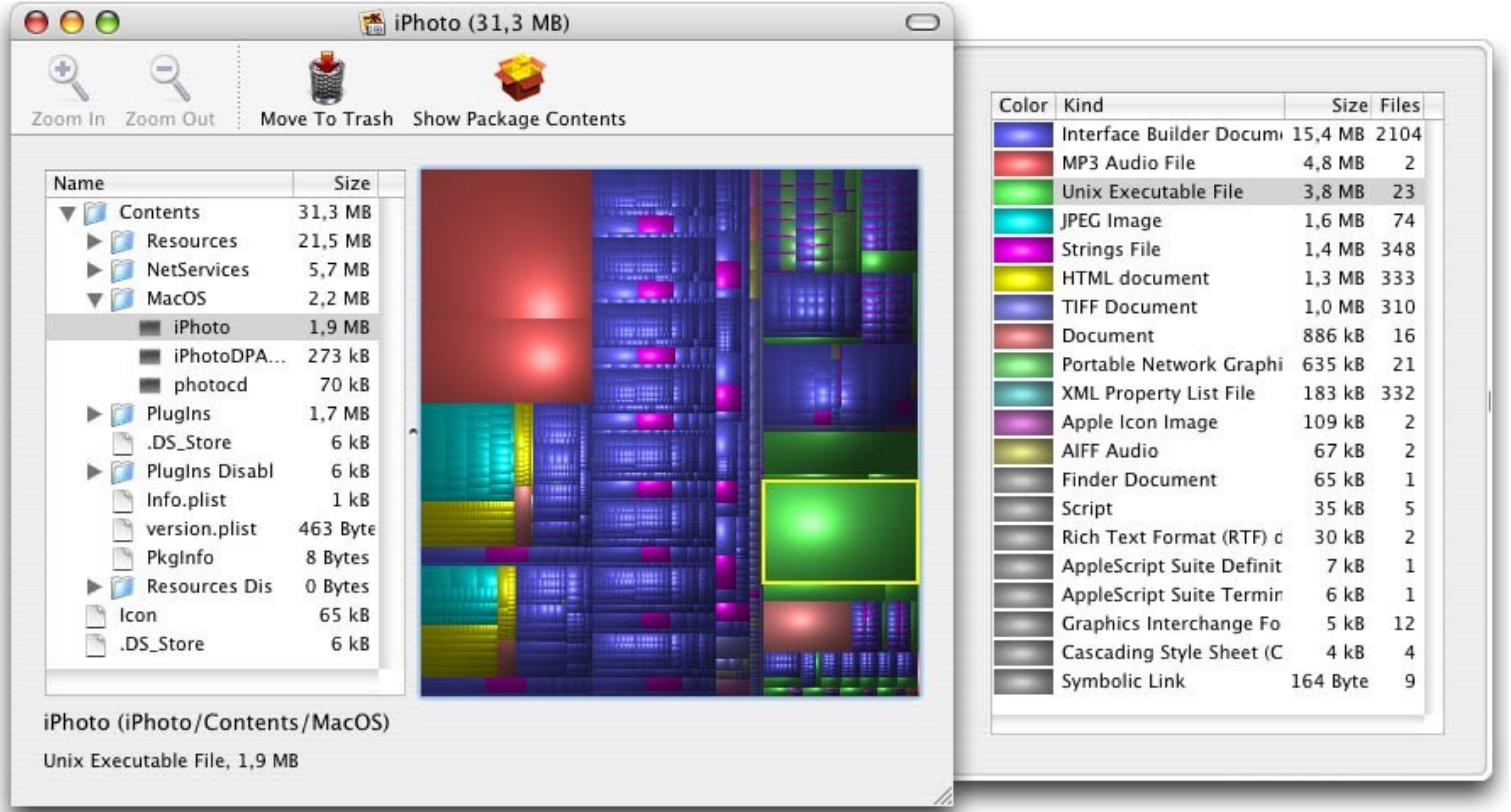
- Leaves are ok, but it can be difficult to find the hierarchy
- Encode this as shading information
- More effective to understand hierarchy





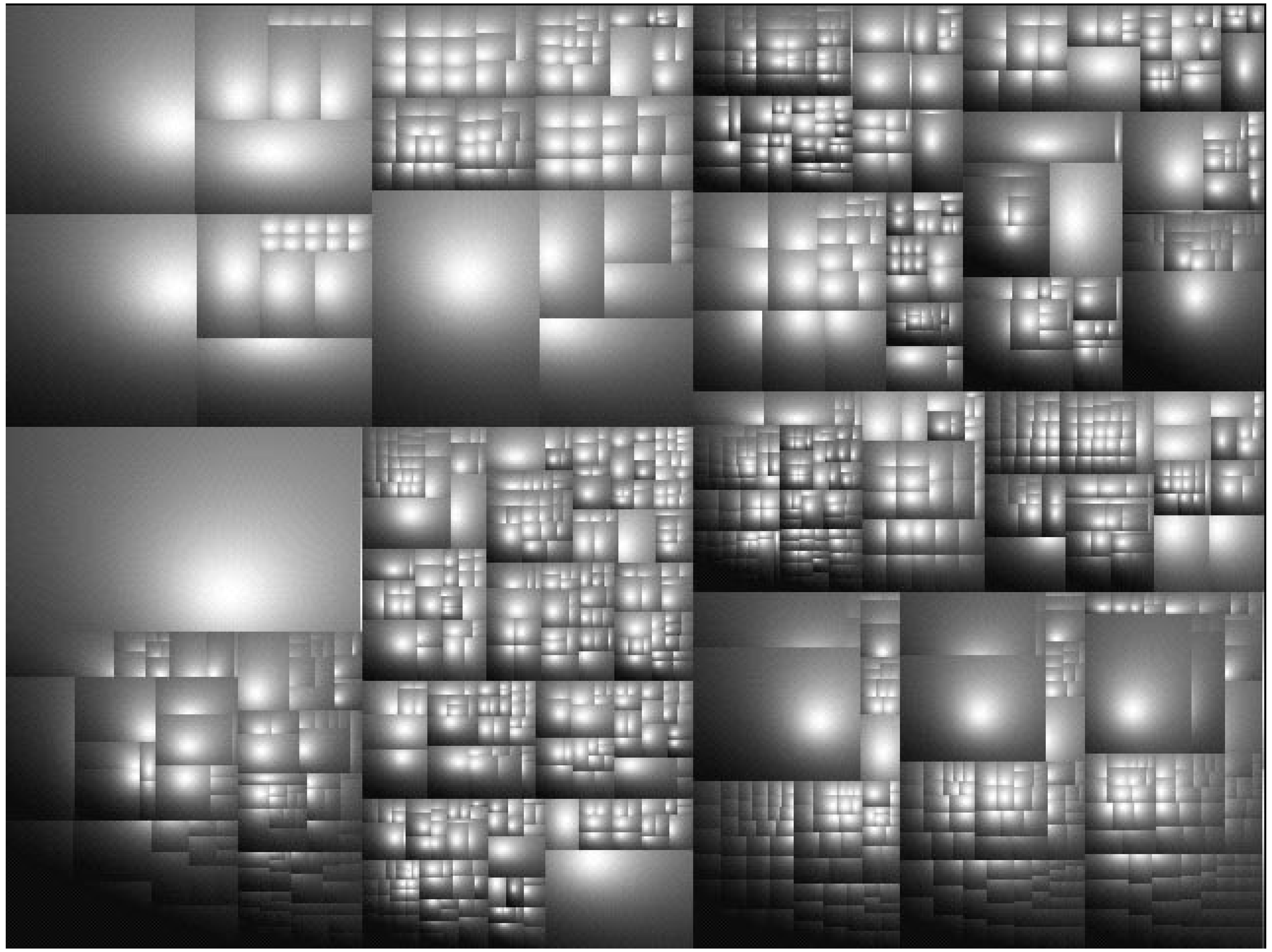
[van Wijk and van de Wetering, 1999]

Disk Inventory

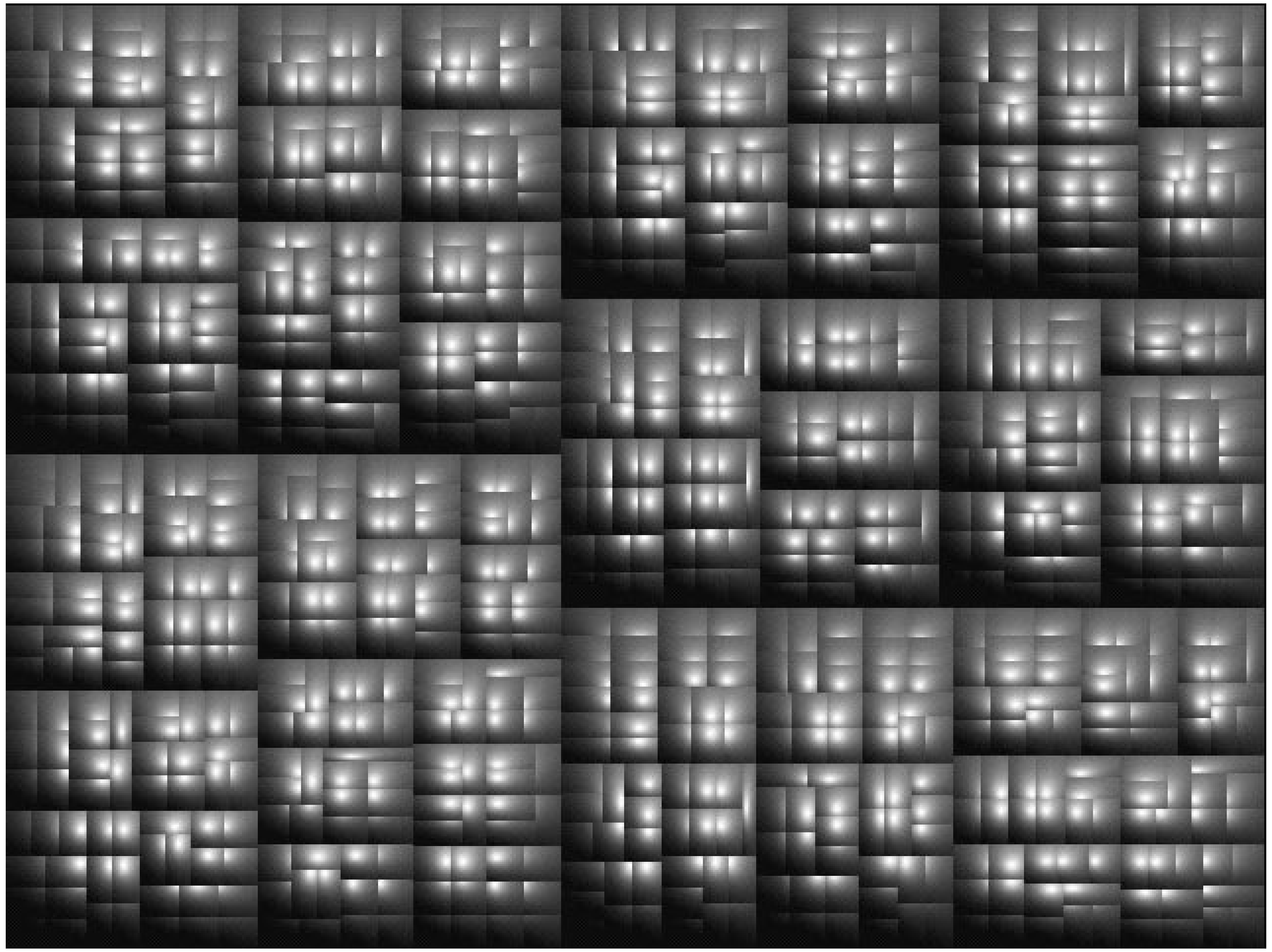


[Disk Inventory X]

Squarified + Cushioned Treemaps



(a) File system



(b) Organization

[Brus et al., 1999]

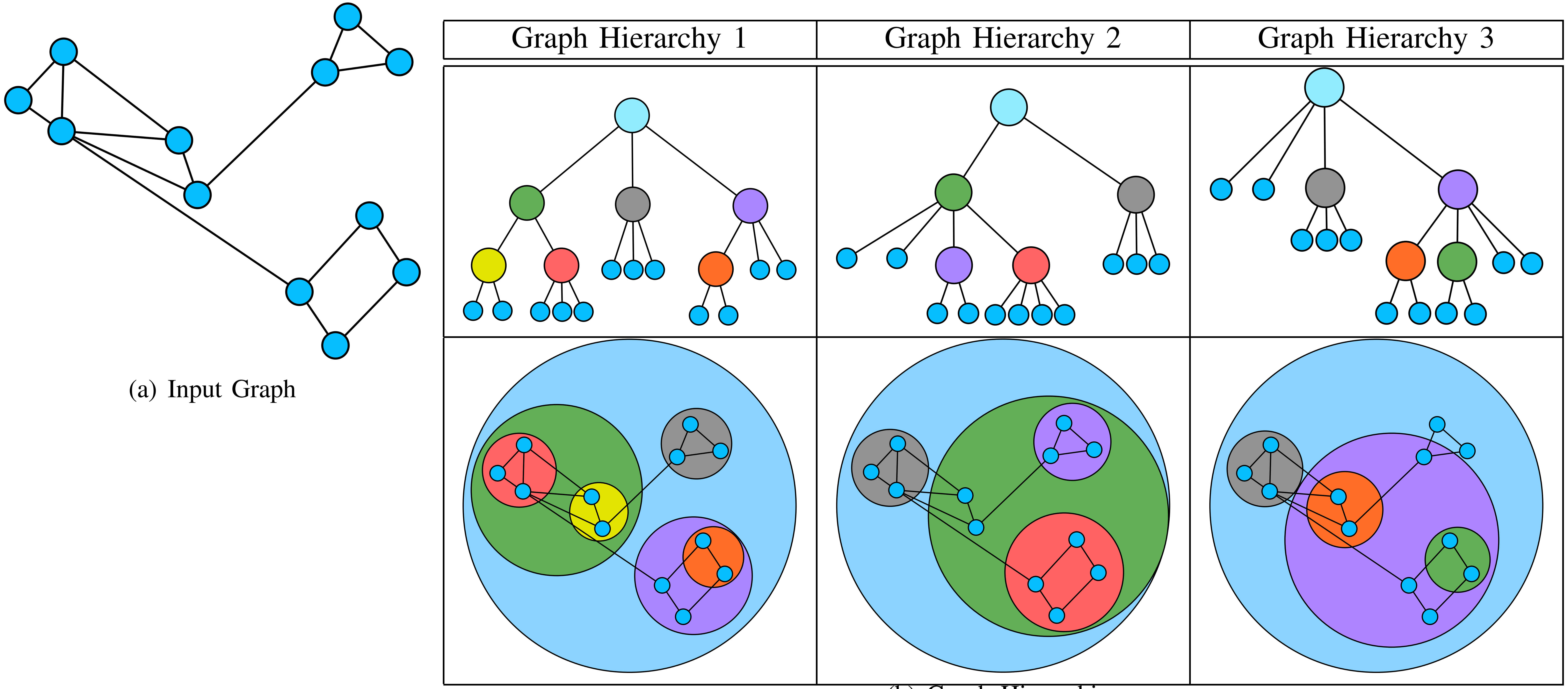
Nested Circles

- Looks more like cluster diagram, but shows hierarchy
- Containment shown by the layering of semi-transparent circles
- Labeling becomes more difficult

[Bostock, 2012]

Compound Networks

- Add a hierarchy to the network (e.g. from clustering)
- GrouseFlocks: uses nested circles with colors



(a) Input Graph

(b) Graph Hierarchies

[Archambault et al., 2008]